FORM PTO-1390 US DEPARTMENT OF COMMERCE REV. 5-93PATENT AND TRADEMARK OFFICE **TRANSMITTAL LETTER TO THE UNITED STATES DESIGNATED/ELECTED OFFICE (DO/EO/US) CONCERNING A FILING UNDER 35 U.S.C. 371**	ATTORNEYS DOCKET NUMBER **P01,0402** U.S. APPLICATION NO. (if known, see 37 CFR 1.5) 10/009397

INTERNATIONAL APPLICATION NO.	INTERNATIONAL FILING DATE	PRIORITY DATE CLAIMED
PCT/EP00/04312	**12 MAY 2000**	**12 MAY 1999**

TITLE OF INVENTION

NETWORK, INTERPRETER FOR SUCH A NETWORK AND METHOD FOR OPERATING A NETWORK

APPLICANT(S) FOR DO/EO/US

ANDREAS HOFSTETTER

Applicant herewith submits to the United States Designated/Elected Office (DO/EO/US) the following items and other information:

1. ☒ This is a **FIRST** submission of items concerning a filing under 35 U.S.C. 371.
2. ☐ This is a **SECOND** or **SUBSEQUENT** submission of items concerning a filing under 35 U.S.C. 371.
3. ☒ This express request to begin national examination procedures (35 U.S.C. 371(f)) at any time rather than delay.
4. ☒ A proper Demand for International Preliminary Examination was made by the 19th month from the earliest claimed priority date.
5. ☒ A copy of International Application as filed (35 U.S.C. 371(c)(2)).
   a. ☒ is transmitted herewith (required only if not transmitted by the International Bureau).
   b. ☐ has been transmitted by the International Bureau.
   c. ☐ is not required, as the application was filed in the United States Receiving Office (RO/US)
6. ☒ A translation of the International Application into English (35 U.S.C. 371(c)(2).
7. ☒ Amendments to the claims of the International Application under PCT Article 19 (35 U.S.C. §371(c)(3))
   a. ☐ are transmitted herewith (required only if not transmitted by the International Bureau).
   b. ☐ have been transmitted by the International Bureau.
   c. ☐ have not been made; however, the time limit for making such amendments has NOT expired.
   d. ☒ have not been made and will not be made.
8. ☐ A translation of the amendments to the claims under PCT Article 19 (35 U.S.C. 371(c)(3)).
9. ☒ An oath or declaration of the inventor(s) (35 U.S.C. 371(c)(4)) - **UNSIGNED.**
10. ☒ A translation of the annexes to the International Preliminary Examination Report under PCT Article 36 (35 U.S.C. 371(c)(5)).

Items 11. to 16. below concern other document(s) or information included:

11. ☒ An Information Disclosure Statement under 37 C.F.R. 1.97 and 1.98; **(PTO 1449, Prior Art, Search Report, 13 References)**.
12. ☐ An assignment document for recording. A separate cover sheet in compliance with 37 C.F.R. 3.28 and 3.31 is included. **(SEE ATTACHED ENVELOPE)**
13. ☒ Amendment "A" Prior to Action and Appendix "A".
    ☐ A SECOND or SUBSEQUENT preliminary amendment.
14. ☒ A substitute specification and substitute specification mark-up.
15. ☐ A change of address letter attached to the Declaration.
16. ☒ Other items or information:
    a. ☒ Submission of Drawing Additions, 3 sheets of drawings, Figures 1-3
    b. ☒ EXPRESS MAIL #EL 843743988 US dated November 12, 2001

U.S. APPLICATION NO. (if known, see 37 C.F.R. 1.5)	INTERNATIONAL APPLICATION NO	ATTORNEY'S DOCKET NUMBER
10/009397	PCT/EP00/04312	P01,0402

17. ☒ The following fees are submitted:	CALCULATIONS	PTO USE ONLY
BASIC NATIONAL FEE (37 C.F.R. 1.492(a)(1)-(5): Search Report has been prepared by the EPO or JPO $890.00 International preliminary examination fee paid to USPTO (37 C.F.R. 1.482) $710.00 No international preliminary examination fee paid to USPTO (37 C.F.R. 1.482) but international search fee paid to USPTO (37 C.F.R. 1.445(a)(2) $740.00 Neither international preliminary examination fee (37 C.F.R. 1.482) nor international search fee (37 C.F.R. 1.445(a)(2) paid to USPTO $1040.00 International preliminary examination fee paid to USPTO (37 C.F.R. 1.482) and all claims satisfied provisions of PCT Article 33(2)-(4) $ 100.00		
ENTER APPROPRIATE BASIC FEE AMOUNT =	$ 890.00	
Surcharge of $130.00 for furnishing the oath or declaration later than ☐ 20 ☐ 30 months from the earliest claimed priority date (37 C.F.R. 1.492(e)).	$	

Claims	Number Filed		Number Extra	Rate		
Total Claims	20	- 20 =	0	X $ 18.00	$	
Independent Claims	03	- 3 =	0	X $ 84.00	$	
Multiple Dependent Claims				$280.00 +	$	
				TOTAL OF ABOVE CALCULATIONS =	$ 890.00	
Reduction by ½ for filing by small entity, if applicable. Verified Small Entity statement must also be filed. (Note 37 C.F.R. 1.9, 1.27, 1.28)					$	
				SUBTOTAL =	$ 890.00	
Processing fee of $130.00 for furnishing the English translation later than ☐ 20 ☐ 30 months from the earliest claimed priority date (37 CFR 1.492(f)). +					$	
				TOTAL NATIONAL FEE =	$ 890.00	
Fee for recording the enclosed assignment (37 C.F.R. 1.21(h). The assignment must be accompanied by an appropriate cover sheet (37 C.F R. 3.28, 3.31). $40.00 per property +						
				TOTAL FEES ENCLOSED =	$ 890.00	
				Amount to be refunded	$	
				charged	$	

a. ☒ A check in the amount of $ 890.00 to cover the above fees is enclosed.

b. ☐ Please charge my Deposit Account No. _____ in the amount of $ _____ to cover the above fees. A duplicate copy of this sheet is enclosed.

c. ☒ The Commissioner is hereby authorized to charge any additional fees which may be required, or credit any overpayment to Deposit Account No. **50-1519**. **A duplicate copy of this sheet is enclosed.**

NOTE: Where an appropriate time limit under 37 C.F.R. 1.494 or 1.495 has not been met, a petition to revive (37 C.F.R. 1.137(a) or (b)) must be filed and granted to restore the application to pending status.

SEND ALL CORRESPONDENCE TO:

SCHIFF HARDIN & WAITE
PATENT DEPARTMENT
6600 Sears Tower
233 South Wacker Drive
Chicago, Illinois 60606-6473

CUSTOMER NUMBER 26574

Mark Bergner
SIGNATURE

Mark Bergner
NAME

45,877
Registration Number

BOX PCT
IN THE UNITED STATES DESIGNATED/ELECTED OFFICE
OF THE UNITED STATES PATENT AND TRADEMARK OFFICE
UNDER THE PATENT COOPERATION TREATY--CHAPTER II

PRELIMINARY AMENDMENT A
PRIOR TO ACTION

APPLICANT(S): Andreas HOFSTETTER

ATTORNEY DOCKET NO.: P01,0402

INTERNATIONAL APPLICATION NO: PCT/EP00/04312

INTERNATIONAL FILING DATE: 12 May 2000

INVENTION: NETWORK, INTERPRETER FOR SUCH A NETWORK, AND METHOD FOR OPERATING A NETWORK

Assistant Commissioner for Patents,
Washington D.C. 20231

Sir:

Applicants herewith amend the above-referenced PCT application, and request entry of the Amendment prior to examination on the United States Examination Phase.

IN THE CLAIMS:

On amended page 27:

replace line 1 with --WHAT IS CLAIMED IS:--;

Please replace original claims 1-20 with the following rewritten claims 1-20, referring to the mark-ups in Appendix A.

1. (Amended) A network for the interconnection of computers, comprising:

a client computer;

a server computer that is configured to store datafiles and transmit them to said client computer when said client computer calls them by sending a corresponding datafile address to said server, wherein said datafiles are structured

to contain both language elements executable at said client as well as language elements executable at said server;

an interpreter in said server configured to interpret and execute said language elements executable at said server;

a further logical or physical system comprising data of a different format than data exchanged between said server and said client; and

a gateway installed at said server and integrated in said interpreter, said gateway being configured to set up a data connection to said further logical or physical system, and configured to automatically convert both incoming as well as outgoing data into appropriate data formats, said gateway configured to be called by language elements of said interpreter.

2. (Amended) The network according to claim 1, further comprising additional gateways which, in addition to said gateway, comprise a plurality of gateways, said plurality of gateways being integrated in said interpreter and being callable by said language elements of said interpreter.

3. (Amended) The network according to claim 1, wherein said interpreter is configured at said server such that said language elements executable at said server are executed at said server after a calling of said datafiles by a client and before transmission of said datafiles to said client.

4. (Amended) The network according to claim 1, wherein said datafile address corresponds to a URL format and wherein said server is a web server, said datafiles being callable with an Internet browser installed at said client.

5. (Amended) The network according to claim 1, wherein said datafiles that are stored at said server and being fetchable by said client correspond to a mark-up

language format that is expanded by said language elements executable at said server.

6. (Amended) The network according to claim 1, wherein said gateway is configured to convert data in a format selected from the group consisting of SNMP, LP, PJMweb, and FTP.

7. (Amended) The network according to claim 1, wherein said gateway is configured to be called by language elements of said interpreter.

8. (Amended) The network according to claim 1, further comprising:

programs configured to drive at least one printer or pre-processing or post-processing devices installed at said server, said programs configured to be called by said interpreter.

9. (Amended) An interpreter for a network, wherein

said interpreter configured to be installed at a server of a network for interconnecting computers, said interpreter being configured to interpret and execute language elements executable at said server that are contained in a datafile stored at said server, wherein a client is configured to receive said datafile, said datafile comprising additional language elements executable at said client.

10. (Amended) The interpreter according to claim 9, further comprising a command for generating string entries in said datafile.

11. (Amended) The interpreter according to claim 9, further comprising a command for setting string entries at a predetermined location of said datafile.

12. (Amended) The interpreter according to claim 9, further comprising a command for reading in a string transmitted from said client to said server, said interpreter being configured to store said string into a predetermined variable.

13. (Amended) The interpreter according to claim 9, further comprising a command for calling a gateway and querying a system connected to said gateway.

14. (Amended) The interpreter according to claim 9, further comprising:

a group of client commands that can be called both proceeding from said client as well as from said server; and

a group of server commands that can only be called proceeding from said server.

15. (Amended) A data carrier comprising an interpreter according to claim 9.

16. (Amended) A method for operating a network for the interconnection of computers having a server and a client, comprising:

storing datafiles on said server that are executable in said server and in said client;

calling said datafiles by said client by sending a corresponding datafile address to said server;

transmitting said datafiles by said server to said client in response to said calling said datafiles by said client;

inquiring by said client to said server, which is a queried server, for a specific service offered by said server, said client using specific parameters of said service;

determining by said queried server whether it can perform said inquired service;

if said server can perform said service, said service performs said service by said server;

if said server cannot perform said service, said server switches said client to a further server or device connected to said network that is capable of executing said service.

17. (Amended) The method for the operation of a network according to claim 16, wherein service offered by said server is executing a print order, and said method further comprising:

forwarding, by said server, said print order to another server or directly to a printer device when said server itself cannot execute said print order.

18. (Amended) The method for the operation of a network according to claim 16, further comprising the steps of:

storing information about said services offered by said server in a databank of said server; and

examining said databank to determine if a service is present for an inquiry by said client.

19. (Amended) The method for the operation of a network according to claim 16, further comprising:

generating an address of a further server or device for said server switching said client to said further server or device; and

communicating said address to said client inquiring said server.

20. (Amended) The method for the operation of a network according to claim 16, further comprising:

installing an interpreter at said server;

interpreting, by said interpreter, language elements executable at said server contained in said datafile;

executing, by said interpreter, said language elements executable at said server;

executing, by said client, language elements executable at said client contained in said datafile.

REMARKS

The present Amendment revises the specification and claims to conform to United States patent practice, before examination of the present PCT application in the United States National Examination Phase. Pursuant to 37 CFR 1.125 (b), applicants have concurrently submitted a substitute specification, excluding the claims, and provided a marked-up copy. All of the changes are editorial and applicant believes no new matter is added thereby. The amendment, addition, and/or cancellation of claims is not intended to be a surrender of any of the subject matter of those claims.

Early examination on the merits is respectfully requested.

Submitted by,

Mark Bergner (Reg. No. 45,877)

Mark Bergner
Schiff Hardin & Waite
Patent Department
6600 Sears Tower
233 South Wacker Drive
Chicago, Illinois 60606-6473
(312) 258-5779
Attorneys for Applicant

CUSTOMER NUMBER 26574

Appendix A
Mark Ups for Claim Amendments

1.[~~Network~~] **(Amended) A network** for the interconnection of computers, [~~whereby at least one computer acts as server (1, 2) and one computer acts as client (3, 4), and~~]**comprising:**

a client computer;

a server computer that is configured to store datafiles [~~stored at the server (1, 2) are transmitted from the server (1, 2)~~]**and transmit them** to [~~the~~]**said** client [~~(3,~~]**computer** [~~4)~~]when [~~the~~]**said** client [~~(3, 4)~~]**computer** calls them by sending a corresponding datafile address to [~~the~~]**said** server[~~ (1, 2)~~], [~~and~~][~~the~~]**wherein said** datafiles **are structured to** contain both language elements executable at [~~the~~]**said** client[~~ (3, 4)~~] as well as language elements executable at [~~the~~]**said** server[~~ (1, 2), and~~]**;**

an interpreter [~~is present at the server (1, 2) for interpretation and execution of the~~]**in said server configured to interpret and execute said** language elements executable at [~~the server, and~~]**said server;**

a further logical or physical system comprising data of a different format than data exchanged between said server and said client; and

a gateway[~~ (12) is~~] installed at [~~the server (1, 2) that can~~]**said server and integrated in said interpreter, said gateway being configured to** set up a data connection to [~~a~~]**said** further logical [~~and/~~]or physical system, [~~whereby the data of the further system comprise a different format than the data exchanged between the server (1, 2) ~~]and [~~the client (3, 4), and the gateway (12)~~]**configured to** automatically [~~converts~~]**convert** both[~~ the~~] incoming as well as [~~the ~~]outgoing data into [~~the ~~]appropriate data formats, [~~and~~][~~whereby the~~]**said** gateway [~~(12) is integrated in the interpreter and can~~]**configured to** be called by language elements of [~~the~~]**said** interpreter**.** [~~(18).~~]

2.[~~ Network~~] **(Amended) The network** according to claim 1, [~~whereby~~]**further comprising additional gateways which, in addition to said gateway, comprise** a plurality of gateways[~~ (12) are~~]**, said plurality of gateways being** integrated in [~~the~~]**said** interpreter and [~~can be called~~]**being callable** by **said** language elements of [~~the~~]**said** interpreter[~~ (18)~~].

3.[~~Network~~] **(Amended) The network** according to claim [~~1 or claim 2, whereby the~~]**1, wherein said** interpreter [~~(18)~~]is [~~fashioned~~]**configured at said server** such [~~at the server (1, 2)~~]that [~~the~~]**said** language elements executable at [~~the~~]**said** server are executed at [~~the~~]**said** server[~~ (3, 4)~~] after [~~the~~]**a** calling of [~~the~~]**said** datafiles by a client and before [~~the~~]transmission of [~~the~~]**said** datafiles to [~~the~~]**said** client[~~ (3, 4)~~].

4.[~~Network~~] **(Amended) The network** according to [~~one of the claims 1 through 3, whereby the~~]**claim 1, wherein said** datafile address corresponds to [~~the~~]**a** URL format and [~~the~~]**wherein said** server[~~ (1, 2)~~] is a web server, [~~so that the~~]**said** datafiles [~~can be called~~]**being callable** with an Internet browser installed at [~~the~~]**said** client[~~ (3, 4)~~].

5.[~~Network~~] **(Amended) The network** according to [~~one of the claims 1 through 4, whereby the~~]**claim 1, wherein said** datafiles **that are** stored at [~~the~~]**said** server [~~(1, 2) ~~]and **being** fetchable by [~~the~~]**said** client[~~ (3, 4)~~] correspond to [~~the format of ~~]a mark-up language **format** that is expanded by [~~the~~]**said** language elements executable at [~~the~~]**said** server.

6.[~~Network~~] **(Amended) The network** according to [~~one of the claims 1 through 5, whereby a respective~~]**claim 1, wherein said** gateway [~~(12) ~~]is [~~provided for the conversion of the~~]**configured to convert** data in [~~one or more of~~]**a format selected from** the [~~following formats:~~]**group consisting of** SNMP, LP, PJMweb, [~~ftp~~]**and FTP**.

7.[~~Network~~] **(Amended) The network** according to [~~one of the claims 1 through 6, whereby the~~]**claim 1, wherein said** gateway [~~or, respectively, gateways (12) are integrated in the interpreter and can~~]**is configured to** be called by language elements of [~~the~~]**said** interpreter[~~ (18)~~].

8.[~~Network~~] **(Amended) The network** according to [~~one of the claims 1 through 7, whereby ~~]**claim 1, further comprising:**

programs [~~for~~]**configured** [~~the~~]**to** drive [~~of ~~]at least one printer [~~and/~~]or pre-processing or post-processing devices [~~are ~~]installed at [~~the~~]**said** server[~~ (1, 2) and these~~]**, said** programs [~~can~~]**configured to** be called by [~~the~~]**said** interpreter[~~ (18)~~].

9.[~~Interpreter~~] **(Amended) An interpreter** for a network[~~according to one of the preceding claims, that can~~]**, wherein**

said interpreter configured to be installed at a server [~~(1, 2)~~]of a network for [~~the interconnection of~~]**interconnecting** computers**, said interpreter being configured to interpret** and [~~is fashioned for the interpretation and execution of~~]**execute** language elements executable at [~~the~~]**said** server[~~ (1, 2)~~] that are contained in [~~datafiles~~]**a datafile** stored at [~~the~~]**said** server[~~ (1, 2)~~], [~~whereby these datafiles by~~]**wherein** a client [~~(3, 4) with the transmission of an address and contain~~]**is configured to receive said datafile, said datafile comprising** additional language elements executable at [~~the~~]**said** client[~~ (3, 4)~~].

10.[~~Interpreter~~] **(Amended) The interpreter** according to claim 9, [~~whereby the interpreter comprises~~]**further comprising** a command for generating string entries in [~~the~~]**said** datafile.

11.[~~Interpreter~~] **(Amended) The interpreter** according to claim [~~9 or 10,~~]**9, further** comprising a command for setting string entries at a predetermined location of [~~the~~]**said** datafile.

12.[~~Interpreter~~] **(Amended) The interpreter** according to [~~one of the claims 9 through 11,~~]**claim 9, further** comprising a command for reading in a string transmitted from [~~the~~]**said** client [~~(3, 4) ~~]to [~~the~~]**said** server[~~ (1, 2) and for storer the~~]**, said interpreter being configured to store said** string into a predetermined variable.

13.[~~Interpreter~~] **(Amended) The interpreter** according to [~~one of the claims 9 through 12,~~]**claim 9, further** comprising a command for calling a gateway and querying a system connected to [~~the~~]**said** gateway.

14.[~~Interpreter~~] **(Amended) The interpreter** according to [~~one of the claims 9 through 13, whereby the interpreter comprises ~~]**claim 9, further comprising:**

a group of client commands that can be called both proceeding from [~~the~~]**said** client as well as from [~~the~~]**said** server[~~,~~]**;** and [~~comprises ~~]

a group of server commands that can only be called proceeding from

[~~the~~]**said** server.

[~~15. Interpreter according to one of the claims 9 through 14, whereby the interpreter is stored on a data carrier.~~]

15. (Amended) A data carrier comprising an interpreter according to claim 9.

16.[~~Method~~] **(Amended) A method** for [~~the operation of~~]**operating** a network for the interconnection of computers [~~that is fashioned according to one of the claims 1 through 8, whereby~~]**having a server and a client, comprising:**

storing datafiles on said server that are executable in said server and in said client;

[~~datafiles stored at servers (1, 2) are transmitted from one of the servers (1, 2) to a client (3, 4) when the~~]**calling said datafiles by said** client[~~(3, 4) calls them~~] by sending a corresponding datafile address to [~~the~~]**said** server[~~ (1, 2) and the servers respectively offers the clients one or more services, whereby,~~]**;**

transmitting said datafiles by said server to said client in response to said calling said datafiles by said client;

[~~given a client~~]**inquiring by said client to said server, which is a queried** [~~inquiry~~]**server,** for a specific service [~~with~~]**offered by said server, said client using** specific parameters [~~on which the~~]**of said** service[~~ is based, the~~]**;**

determining by said queried server [~~determines~~]whether it can perform [~~the~~]**said inquired** service[~~ and, when the server determines that it~~]**;**

if said server can perform said service, said service performs said service by said server;

if said server cannot perform [~~the~~]**said** service, [~~it~~]**said server** switches **said client to** a further server or [~~to a~~]device connected to [~~the~~]**said** network that [~~can execute the~~]**is capable of executing said** service[~~ to the client~~].

17.[~~ Method~~] **(Amended) The method** for the operation of a network according to claim 16, [~~whereby one of the services~~]**wherein service** offered by [~~the~~]**said** [~~servers~~]**server** is [~~the execution of~~]**executing** a print order, and **said method further comprising:**

[~~the~~]**forwarding, by said** server[~~ forwards the~~]**, said** print order to [~~a other~~]**another** server or directly to a printer device when [~~the~~]**said** server itself cannot execute [~~the~~]**said** print order.

18. **(Amended) The** method for the operation of a network according to claim [~~16 or 17, whereby~~]**16, further comprising** the [~~server comprises a data bank in which~~]**steps of:**

storing information about [~~the~~]**said** services offered **by said server** in [~~the network are stored, so that, given~~]a [~~client query, a determination can be made on the basis~~]**databank** of [~~these data banks as to whether the desired~~]**said server; and**

examining said databank to determine if a service is present [~~in the network~~]**for an inquiry by said client**.

19.[~~Method~~] **(Amended) The method** for the operation of a network according to [~~one of the claims 16 through 18, whereby the~~]**claim 16, further comprising:**

generating an address of a further server or device for said server switching **said client** to [~~a~~]**said** further server or [~~to a~~]device[~~connected to the network is implemented by generating the address of the further server or of the device~~]**;** and[~~ by~~]

communicating [~~the~~]**said** address to [~~the querying~~]**said** client **inquiring said server**.

20.[~~Method~~] **(Amended) The method** for the operation of a network according to [~~one of the claims 16 through 19,~~]**claim 16, further** comprising**:**

installing an interpreter [~~according to one of the claims 9 through 15.~~]**at said server;**

interpreting, by said interpreter, language elements executable at said server contained in said datafile;

executing, by said interpreter, said language elements executable at said server;

executing, by said client, language elements executable at said client contained in said datafile.

Document comparison done by DeltaView on Monday, November 05, 2001 16:35:12

2/pts

SPECIFICATION

TITLE

NETWORK, INTERPRETER FOR SUCH A NETWORK AND METHOD FOR OPERATING A NETWORK

BACKGROUND OF THE INVENTION

FIELD OF THE INVENTION

[0001] The invention is directed to a network, to an interpreter for such a network and to a method for operating a network. In particular, the invention is directed to a network for a printing system.

DESCRIPTION OF THE RELATED ART

[0002] As a rule, a number of printers and printing systems and potentially corresponding pre-processing and post-processing devices that, for example, cut the printed paper to a specific format or bind it are connected to existing data networks such as, for example, LANs (local area networks) and WANs (wide area networks). The individual printers and printing systems differ greatly in their performance capability; for example, ink jet printers with a print output of 4 pages per minute or high-performance printers with a print output of 40 pages or more per minute can be connected to the data network. The printers and printing systems differ not only in terms of their printing output but also in terms of their printing quality. For example, although there are printers that print only in a single color (monochromatically), color printers are also being increasingly utilized. Electrophotographic high-performance printer devices are currently in the position of printing two colors without further effort, i.e., in a "spot color mode" of highlight color mode. Such a printer, for example, is known by the name PAGESTREAM 200DSC of Océ Printing Systems GmbH.

[0003] The different printers and printing systems are usually also connected to the respective network with different system interfaces such as, for example, the SNMP (Simple Network Management Protocol) or the DMI (Desktop Management

Interface). Although there is often a physical connection to a plurality of printers or printing systems via the network, only a part of the printers can be addressed.

[0004] Given the rapid development of the Intranet and Internet, with which several LANs and WANs are connected to a single network (particularly as a result of the introduction of the WWW service based on the HTTP protocol (Hypertext Transport Protocol)), the number of printers and printing systems that can be fundamentally addressed is increasing explosively, with a corresponding increase in the number of protocols for addressing the printers and printing systems.

[0005] This success of the Internet and of the Intranet was greatly promoted by the introduction of the World Wide Web (WWW) that can be simply operated by Internet users and allows the widest variety of information to be fetched from the greatest variety of different locations in the entire world. The WWW service of the Internet is based on the client-server principle. The communication ensues between a web server, (or "WWW server") which offers information, and a client that displays the information. The information are stored on the web server in pages, in which the data in the pages are stored in, for example, the "HTML format" (Hypertext Markup Language). Other formats are also in use, such as XML. These formats are derivatives of the underlying SGML format (Standard General Markup Language). These formats are referred to as markup languages since the make-up of the language can be described and defined with them.

[0006] A corresponding document is composed of normal text in which control instructions ("tags") are inserted into the text. Among other things, these tags influence the layout that is displayed later in the observation program, the browser, at the client. For example, there are tags to generate headings or tags that can modify the print image. The tags are always enclosed in the bracketing "< ... >".

[0007] The transmission of information from the web server to the client ensues according to the HTTP protocol; this information transmitted to the client is read by its installed browser and the individual tags are interpreted so that the information is displayed at the picture screen of the client in the predetermined fashion.

[0008] Since essentially only static images and text that can be displayed in the HTML format, the WWW service has been augmented by JAVA for mixing in

multimedia elements, animations or programs. JAVA is an object-oriented programming language for web applications developed by SUN Microsystems. JAVA supports text, hypertext, graphics, audio and animation functions.

[0009] The program packets programmed in JAVA can either be loaded onto the client from the web server as "JAVA applets" or can be pre-installed at the client as "JAVA applications". A modification of JAVA that is referred to as JAVAscript allows the insertion of program packets in the HTML page.

[0010] JAVA applications with which both administration functions at the printers and printing systems as well as the sending of print orders to the individual printers or printing systems can be implemented have been developed in order to drive printers and printing systems via the Internet. These JAVA programs represent significant progress over the previous situation since they make it possible for the user of a client computer to drive a plurality of printers and printing systems via the Internet independent of the platform used. The type of printing systems, i.e., the interface types, that can be driven by a client are determined by the respective JAVA application, in which a separate program element is provided for each interface type. This means that given the large variety of interfaces that are driven with such a JAVA application, JAVA application must be all the more extensive . The same is true of the function scope that is driven given such a JAVA application. This produced JAVA applications that are extensive programs with, for example, a data scope of up to 9 MB or greater. They thus make use of considerable computer power and memory space at the client and are therefore only expedient for users who wish to regularly drive different printers and printing systems.

[0011] It is also known to provide programs executable at the server that can be called by the client. These programs are deposited at the server either as completely compiled programs or as program code that can be interpreted by an interpreter. The programming language Perl is often employed for Internet applications for such an interpretable program code. It is especially suited for data bank applications to be executed at the server. However, pages that can be loaded from the server to the client can also be generated with Perl; Perl provides commands with which the make-up of corresponding pages can be constructed. The Perl program code can also generate HTML program code. When such a

program is called at the server, the complete program code is interpreted and executed at the Server.

[0012] The publications of Jörn Heid, "Kettenreaktion" in iX 11/1998, pages 166-171, of Jörn Heid, "Was es sein darf" in iX 11/1998, pages 64-67 or of Rainer Klute, "Mehr als Applets" in iX 11/1998, pages 60-63 describe servlets that are Java programs stored in a server. The servlets can be linked in HTML pages and are executed at the server when one of these pages is called. For example, parts of or complete HTML pages can be generated with such servlets. Servlets thus represent a possibility for linking Java programs executable at the server into HTML pages.

[0013] The publication of Jörn Heid, "Hand angelegt" in iX 11/1998, pages 68-70 discloses "Java server pages" (JSP) in which script code and HTML code can be contained in the same document, where the scripts are replaced by their results before being sent to the client. This technique makes it possible to provide datafiles at the server that are called by the client and comprise language elements that can be executed both at the client as well as at the server. Similar techniques are known under the trademarks Live Wire™ and Active Server Pages™.

[0014] The publication by B. Merkleund, F. Pilhofer, "Perlin vor die Middleware" in iX 4/1999, pages 154-165 relates to the CORBA mapping for script languages. CORBA (Common Object Request Broken Architecture) is a complicated architecture for a query and communication service. All possible objects within a network can be communicated with CORBA; thus it is also possible to communicate objects provided on periphery devices with CORBA. To this end, however, it is necessary that these objects are fluent in a CORBA-specific protocol for communication. Furthermore, CORBA comprises an API (Application Program Interface) with which individually fabricated programs at the server can communicate with the CORBA system. These individual programs can also be provided for the communication of periphery devices. It is also possible to map CORBA-independent protocols onto the CORBA protocol. This involves the production of such a mapping or of independent programs, as well as the generation of CORBA objects at periphery devices since communicating objects requires adherence to the protocol prescribed by CORBA, which is very extensive and complicated due to its universal applicability.

[0015] German patent document DE 197 04 694 A1 discloses a method and an apparatus for controlling a periphery device via the Internet that employs known CGI scripts, which are programs deposited at the server independently of an HTML page but that can be called from HTML pages with a corresponding reference and can be executed at the server.

[0016] The Druckerbuch of Océ Printing Systems GmbH, ISBN 3-00-001019-X, 3rd Edition, October 1998, discloses a print server for professional, Internet-independent applications on pages 12-2 through 12-8. This print server is called PRISMApro7. Such a print server is composed of a high-performance personal computer with corresponding software in order to be able to drive one or more printers, particularly a fast printer or high-performance printer. Such a print server is linked into a network and converts the incoming data streams into corresponding print data. Depending on the embodiment, the print server can process data in the formats AFDPS, Line/SF, PostScript, TIFF, PDF, LCDS, Line/O and PCL. Such print servers are especially utilized in data bank applications in which a great quantity of data with variable data is printed from a data bank.

[0017] Furthermore, pages 14-2 through 14-12 in the Druckerbuch of Océ Printing Systems GmbH, ISBN 3-00-001019-X, 3rd Edition, October 1998, describes the print production control system Océ Domain7. In this system, one or more printers with pre-processing devices and post-processing devices are connected via a network. This controls and monitors the print production. This system provides a network-based client-server solution with a data bank that is installed on a central data bank server. All machine and operating data acquired in the production process are collected in this data bank and are offered to the various clients for later evaluations or can be exported into client-specific applications. Interfaces to the high-performance printers connected in the network and to the various hardware and software components are realized with Océ Domain7. The industry standards DMI (Desktop Management Interface), LMO (Large Mailroom Operations) and ODBC (Open Data Base Connectivity) are supported. Océ Domain7 is a high-performance control and monitoring system that, in particular, is employed in printing centers wherein high-performance printers are coupled to devices for pre-processing and for post-processing.

[0018] International patent document WO 99/18534 discloses a method with which queries from different servers can be processed in a computer network in which the processing of the queries is automatically divided among the servers. To this end, the servers comprise a "load balance module" with which the respective work load of the servers connected via the network is determined and the query is assigned to that server that exhibits the lowest work load. This method is particularly provided for employment in the Internet.

[0019] European patent document EP 0 874 306 A2 discloses a network printing system to which a plurality of clients can be connected, these clients being capable of executing print orders at different printers via print servers provided in the network. The print servers are fashioned with a layer architecture and comprise a communication interface. This communication interface can be provided with known program packets that automatically translate the print format of the print order into a print format suitable for the printer.

[0020] European patent document EP 0 872 792 A2 discloses a network for communication with printing systems that is based on the Internet. In particular, functions are provided in order to correctly print out HTML datafiles that contain a reference to a further image datafile. To this end, an interpreter that executes the HTML datafile and can insert the corresponding image datafiles into the HTML datafiles is provided at the output device that is connected to the network. Languages/formats, such as JPEG and GIF, that describe image datafiles are interpreted with these interpreters. Alternately, it is also possible that a translation program for the translation of the image datafiles allows the provision of a directly executable program code provided in the network instead of the interpreter.

[0021] International patent document WO 86/29663 discloses HTML datafiles into which executable scripts are linked. Such a script is executed at the web server and, for example, is programmed in an interpreter language such as Basic or Tool Controlling Language or in a compiler language such as "C" and is compiled in a correspondingly runnable program. This document also discloses how such HTML datafiles can be produced and deposited at a server.

SUMMARY OF THE INVENTION

[0022] The invention is based on the object of finding a simple technical solution for the drive of different printers and printing systems.

[0023] This object is achieved by a network for the interconnection of computers, comprising a client computer; a server computer that is configured to store datafiles and transmit them to the client computer when the client computer calls them by sending a corresponding datafile address to the server, wherein the datafiles are structured to contain both language elements executable at the client as well as language elements executable at the server; an interpreter in the server configured to interpret and execute the language elements executable at the server; a further logical or physical system comprising data of a different format than data exchanged between the server and the client; and a gateway installed at the server and integrated in the interpreter, the gateway being configured to set up a data connection to the further logical or physical system, and configured to automatically convert both incoming as well as outgoing data into appropriate data formats, the gateway configured to be called by language elements of the interpreter.

[0024] This object is also achieved by an interpreter for a network, wherein the interpreter configured to be installed at a server of a network for interconnecting computers, the interpreter being configured to interpret and execute language elements executable at the server that are contained in a datafile stored at the server, wherein a client is configured to receive the datafile, the datafile comprising additional language elements executable at the client.

[0025] This object is also achieved by a method for operating a network for the interconnection of computers having a server and a client, comprising storing datafiles on the server that are executable in the server and in the client; calling the datafiles by the client by sending a corresponding datafile address to the server; transmitting the datafiles by the server to the client in response to the calling the datafiles by the client; inquiring by the client to the server, which is a queried server, for a specific service offered by the server, the client using specific parameters of the service; determining by the queried server whether it can perform the inquired service; if the server can perform the service, the service performs the service by the server; if the server cannot perform the service, the server switches the client to a

further server or device connected to the network that is capable of executing the service.

[0026] The network of the invention is a network for the association of computers that connects at least one client to at least one server, in which datafiles stored at the server can be called by the client by communicating a datafile address, resulting in a corresponding datafile being transmitted to the server. These datafiles capable of being transmitted from the server to the client contain language elements that are executed at the client. An interpreter is provided at the server that can interpret and execute further language elements executable at the server that are contained in the datafiles stored at the server and fetchable by the client.

[0027] With the invention, datafiles are provided at the server that contain both language elements executable at the server as well as language elements executable at the client. This results in significant advantages, since it is no longer necessary -- as known, for example, for Perl programs executable at the server -- to generate language elements executable at the client via language elements of a different language, which involves considerable programming outlay.

[0028] Inventively, a gateway is integrated into an interpreter of a server in which language elements for calling the gateway are provided. This permits a computer language with which gateways can be directly called being offered at the server. The integration of the gateway into the programming language allows the direct drive of arbitrary periphery devices from the programming language. As a result of this, a user, who produces a datafile to be stored at the server, can handle the control of the periphery devices simultaneously with the production of this datafile. This is not possible in traditional systems since the corresponding interfaces can only be deposited at the server as CGI script, servlet, or the like by way of programs that can be produced by specialists or must be mapped in an involved way at a known broker system.

[0029] Compared to the above-described CORBA system, the invention can be realized significantly more simply and with significantly less program code in the server and offers the user significantly more possibilities in the definition of his applications since it can be freely programmed as programming language.

[0030] An inventive network thus allows any user who has simple auxiliaries available for producing such datafiles stored at the server to also drive periphery devices that are in communication with the server either directly or via the network. In particular, printers can be driven in a simple way.

[0031] When the language elements executable at the client correspond to a mark-up language (such as, SGML, HTML, XML), the datafiles can be edited with most standard editors and text processing programs. No knowledge of the mark-up language is often needed for producing simple mark-up datafiles since the corresponding syntax is automatically inserted by the editor or by the text processing program. Complex applications can thus be produced with the invention using text processing programs that are simple to use. The language elements executable at the server must merely be attached into the familiar datafiles fetchable from the server. The combination of language elements executable at the client as well as executable at the server in one datafile thus assures a significant facilitation for the user who establishes the datafiles at the server since he can produce these datafiles using familiar mechanisms and need not encode the commands executable at the client in another computer language upon whose execution they are artificially generated.

[0032] By providing an interpreter, which investigates the datafiles called by the client for language elements contained within and potentially interprets and executes these language elements at the server, the execution of an inherently arbitrary program at the server can be initiated by the transmission of only a single address from the client to the server. This creates the possibility of being able to call arbitrary programs previously deposited in corresponding datafiles at the server with minimal programming outlay at the client and absolutely minimal data volume that must be transmitted between the server and the client. These programs can be used by a plurality of clients, so that a corresponding program packet need be deposited at only a single location in the network, i.e., at the server.

[0033] The advantages achieved with the invention become especially clear when the data transmission in the network is based on a standardized data transmission such as that according to the HTTP protocol that is applied in the Intranets and Internets. Given employment of such a standard, only a traditional

browser without further, additional programs is required at the client in order to drive the corresponding functions. Up to now, involved programs, for example in JAVA, were loaded onto the clients resulting in only the desired functions onto which the corresponding programs were installed or loaded being driven at the clients. Given the inventive network, in contrast, every client at which a browser is provided can call the corresponding programs at the server without additional programs having to be loaded onto the client. Only an access authorization to the respective server is required.

[0034] This provides a critical functional advantage of the invention compared to the traditional system in that it is possible according to the invention that program elements that automatically switch the client to a further server are contained in the datafiles stored at the server. This switching ensues, for example, simply by transmitting the address of the further server from the original server to the client, this then being executed at the client, i.e., being sent to further servers resulting in a connection being set up between the further server and the client.

[0035] This switching function can also be designed as an automatic switching function in which a datafile is created at the server that contains the services of other servers, so that the client is automatically switched to another server that can provide the service when a client asks for a specific service at this one server and this server cannot provide the service. This functions appears to the client as though his order were being negotiated between the servers in order to identify a particular server that can fill the order. This function of automatic forwarding is therefore also called "trading".

[0036] The trading also comprises the possibility that the server functions as a type of transit or relay station. In this role, the server switches a connection between the client and a further workstation such as a further server of a printing station or the like, where the server is connected between the workstation and the client and correspondingly forwards the data. Since the server is provided with a gateway, the data transmission between the further workstation and the server and between the server and the client can be based on a different system or different protocol, and where the server correspondingly translates the data so that a friction-free/transparent data transmission is possible.

[0037] When corresponding datafiles are deposited at the server, a printing system with such a client-server network allows every client who can access this server to address the printer devices drivable with the server and pre-processing and post-processing devices without specific software for this having to be installed at the client. The individual print orders can also be forwarded from a server to another server in order, for example, to be printed out by a printer that is especially suited for this specific print order. The server can be provided with gateways for the conversion of the data into specific print protocols or other types of transmission protocols as well, so that printers coupled to the network in the greatest variety of ways can be driven with a single server.

[0038] A print station composed of at least one printer with a potential pre-processing and post-processing device can be coupled to the network in a simple way in that the devices of the print station (printer, pre-processing and post-processing devices) are connected to a server that is provided with an inventive interpreter and at which corresponding programs for the drive of the individual devices of the print station are stored in datafiles that can be fetched by the client. The devices of such a print station can then be driven by any arbitrary client via the network insofar as there is a corresponding access authorization to the server. The invention thus creates a possibility with which a network access to the print station that can be addressed with extremely simple technical mechanisms merely by attaching a server to an existing print station.

[0039] The inventive network is fashioned for the drive of printers and corresponding pre-processing and post-processing devices. However, it can also be employed for the drive of any other devices since the basic principle of the present invention, providing an interpreter at a network server that interprets datafiles called by the client, can be transferred to further, arbitrary network applications.

DESCRIPTION OF THE DRAWINGS

[0040] The invention is explained in greater detail below by way of example with reference to the attached drawings.

Figure 1 is an inventive network in a block circuit diagram, shown schematically;

Figure 2 is an interface query in a flowchart that shows the program execution of the computer program attached as an appendix; and

Figure 3 is a trading method in the flowchart.

DETAILED DESCRIPTION OF THE INVENTION

[0041] Figure 1 shows an exemplary embodiment of an inventive network. The network comprises a first web server 1, a second web server 2 and a first and second client 3, 4. The first and second web server and the first client are connected to one another via data lines 5, 6 via a LAN, where the data line 5 connects the first web server to the first client and the data line 6 connects the first web server 1 to the second web server 2. An Intranet is operated on the LAN.

[0042] There is no fixed, physical data connection between the first web server 1 and the second client 4. A corresponding data connection 7 (for example, via the telephone network) is set up as needed, and the Internet is employed as a transmission medium.

[0043] The first client 3 is, for example, an office computer of a user of the inventive network connected to the first web server 1 via the Intranet; in contrast, the second client 4 is a personal computer at the user's home with which the user can dial in with a modem in the Internet and set up a connection to the first web server 1. Since the data connection 7 is not permanently present, it is shown with broken lines in Figure 1.

[0044] A respective browser is installed at both clients 3, 4, so that the two clients 3, 4 can communicate with the services known from the Internet with the web server 1. For example, these services include Telnet (that enables a terminal simulation) and/or FTP (File Transfer Protocol, with which datafiles can be transmitted). The present exemplary embodiment employs the service of the World Wide Web (WWW) that can automatically utilize further services of the Internet such as, FTP, News, Telnet, Gopher, E-mail, etc. Two devices 8, 9 are connected to the first web server 1, where the device 8 is connected to the web server 1 via a serial line (RS 232/V24) 10, and a connection 11 according to the SNMP protocol (Simple Network Management Protocol) is installed between the device 9 and the server 1. The SNMP protocol is particularly employed for driving devices connected to a

network. A gateway 12 is provided at the web server 1, this gateway converting the data incoming via the SNMP connection 11 into the Internet protocol applied by the web server 1 on the data connections 5 through 7 or converting the data in the opposite direction according to the Internet protocol onto the SNMP protocol when they are transmitted from the web server 1 to the control device 9.

[0045] All hardware and software components at the web server that set up a connection to a communication system or network different from the Internet are a gateway in the sense of the invention. A plurality of gateways to other communication systems can be provided at the web server 1 that, for example, are based on the DMI standard (Desktop Management Interface), the LP standard (Line Printer), SLP (Service Location Protocol), and/or the IPP standard (Internet Printing Protocol). Gateways to the PJMweb are possible. PJMweb is a web-based print client that has been developed by Océ. A gateway can also generate a terminal emulation. In particular, gateways can be provided that are fashioned for the drive of specific printer communication systems.

[0046] Since a connection to different networks can be produced via the gateways, an arbitrary plurality of devices can be fundamentally connected to a server provided with a gateway or can be driven by the server.

[0047] The second web server 2 is connected to a further device 13 via a line 14. To this end, a communication program 15 specifically fashioned for the device 13 is installed at the second server 2. This communication program 15 is a compiled program for "web-based management". Such programs are respectively configured for a specific application such as driving the device 13, resulting in only a single connection to the device 13 being set up and only a specific device type being driven with this program.

[0048] In addition to other service programs, the first and second web server 1, 2 comprise a WWW service program 16 that comprises the functions known from the Prior Art, i.e., that, upon receipt of a URL, it reads the corresponding pages out from a data store 17 of the web server 1, 2 and sends them to the client 3, 4 that sent the URL. Inventively, an interpreter 18 is provided at the server 1, 2 with which the pages deposited in the store 17 are interpreted before being sent to a client 3, 4, i.e., that language elements contained in the pages are executed by the interpreter

18. The individual pages are stored in the HTML format, i.e., they comprise standardized language elements that can be executed by the browser of the client 3, 4. The language elements executable by the interpreter 18 are independent of language elements executable by the browser. They can also be viewed as a supplement to the HTML format, for which reason the format of the stored pages can also be referred to as "expanded HTML" format.

[0049] A program code of an exemplary program explained in greater detail below is illustrated in the appendix. A brief description of the commands used follows, although the specific use of these specific commands is exemplary. Like most computer languages, the interpreter comprises commands for string processing, system commands and structure commands. As needed, the interpreter can be provided with further groups of commands. In addition to the previously mentioned groups of commands, the interpreter comprises a command with which new commands of the interpreter can be generated. This command is known as "add function" with which an arbitrary command (function) can be constructed from arbitrary commands available on the server (machine language, operating systems or other standard language). "add function" is not employed in HTML pages, but can be called by the user in a program development environment.

[0050] Important string processing commands are "userDefineString", "userGetCGIString", "userPreReplaceString", "userPostReplaceString" and "userCompose". String variables can be defined with "userDefineString". A string input at the client can be read with "userGetCGIString". The commands "userPreReplaceString" and "userPostReplaceString" serve for arranging strings at a predetermined location of the page, where the string is arranged at the corresponding location immediately after the execution of the command given the command "userPreReplaceString". This contrasts with where the string is only arranged at the predetermined location after the processing of all language elements given "userPostReplaceString". The string represented by the command "userPreReplaceString" can thus still be modified by the further language elements to be executed by the interpreter, in contrast to where the string arranged with the command "userPostReplaceString" is placed at the end, resulting in an unmodifiable string.

[0051] One of the most powerful system commands is "userSystem (...)", with which a command of the operating system of the server or of a further program installed on the server can be provided as a parameter. As a result, commands and programs established at the server can be called with the interpreter. This command is called from an HTML page.

[0052] A sub-group of system commands are control commands that, for example, serve for the drive of a specific printer. These commands usually correspond to the respective printer system commands that are respectively converted as a command of the interpreter and can be supplemented by further, higher control commands.

[0053] The structure commands serve for producing a program structure with branchings, loops and the like. Corresponding structure commands are, for example, "userFor", "userGoSub" or "userIf". Corresponding to the syntax of HTML, the commands executable by the interpreter 18 are also placed in brackets "<...>".

[0054] Given the commands of the interpreter 18, a distinction is made between two classes, namely one with commands that cannot be directly called by the client and a further class whose commands can be directly called by the client. Of the commands recited above by way of example, those that begin with "user" cannot be called by the client. This division into two classes of commands serves for security since, if a user of an arbitrary client could generate a "userSystem" command on the server, he could manipulate the server at will. In order to avoid such a manipulation, the store 17 is subdivided into a free memory area 17a and a locked memory area 17b. In the free memory area, the client can call pages and these are potentially supplied with parameters; in contrast, the locked memory area 17b contains pages that do not receive parameters directly from the client or that cannot be called by the client. These pages are only called by the interpreter.

[0055] The interpreter monitors the parameter handover and the program execution. The interpreter 18 is fashioned such that security-relevant commands are only executed when they are stored in the locked memory area 17b. A user at one of the clients can merely read the content of the free memory area 17a and directly address it. The pages deposited in the locked memory area 17b can only be addressed indirectly via the interpreter. The administrator of the server can deposit

control programs needed for the drive of the devices 8, 9 and 13 in the locked memory area 17b of the server, these control programs usually being security-relevant.

[0056] The program code listed in the appendix is explained in greater detail below, and its execution is illustrated by the flowchart in Figure 2. In the initialization step S1 "ini", specific values and strings are set or defined for respective querying servers.

[0057] The SNMP gateway 12 is called with step S2, where the value "1" indicates that this gateway is supposed to read something in, and the result is stored in "SNMPVALUE". Depending on a user input, an address of a data bank entry (a managed object of an MIB) is queried. The result of this query represents the computer type that is addressed via the SNMP interface.

[0058] Based on the value stored in the variable "SNMPVALUE", a check is carried out in the next step S3 to see whether the operating system of the computer addressed via the SNMP gateway is Microsoft Windows. When the operating system is Windows, the program execution branches to the query S4 where an SNMP value is read in anew and a check is carried out on the basis of this value to see whether a predetermined printing system ("Imagestream") runnable under windows is established at the computer. When the query S4 indicates the presence of such a printing system, this is stored in step S5.

[0059] After the storing event in step S5 or if one of the queries S3 or S4 was answered in the negative, the program execution changes to step S6 with which a check is carried out to see whether the operating system of the computer addressed via the SNMP gateway is a UNIX operating system. When the query indicates that the operating system is a UNIX operating system, the program execution branches to step S7 with which an SNMP is read in again, and a check is carried out to see whether a UNIX printing system (PJM or Prisma) is established on the computer. When the query yields such a printing system, then this is stored in step S8.

[0060] After the storing event of step S8 or when one of the two queries S6 and S7 had a negative outcome, the program execution switches to a step S9 with which a check is carried out to see whether the computer addressed via the SNMP gateway is a computer that can be driven at all with the SNMP protocol. When the

query indicates that the computer can fundamentally be driven with the SNMP protocol, then a PING query is called in step S10 and the result of the PING query is checked in step S11 to see whether the computer is operating (online). When this query S11 indicates that the computer is operating (online), then this is stored in step S12; in contrast, when the computer is not operating (offline), then this corresponds to the default setting, so that no additional storing is necessary.

[0061] The PING query is a program module established at many servers that is not only present in the Internet. It is therefore called with the "userSystem" function already described above or is integrated into the interpreter as an independent command. A renewed programming of this program module is thus superfluous.

[0062] After the storing event of step S12 or when the query in step S9 indicated that the computer is an SNMP computer or when the query in step S11 yielded that the computer was not operating, the program execution switches to step S13 with which an image presenting the printing system is registered. This ensues with the command "userCompose" with which the variable "SNMPVALUE" documented by the above queries and storing events is evaluated (see the line under the command "userCompoase"), where the result is "exclusive". This is prescribed in the following line by the value "false". This means that only a single image can be read into the variable "REPLACEMENT". Dependent on the above result, an image for an Imagestream printing system (imagestream.gif), a Prisma printing system (Prisma.gif), an image for the non-operational status of the computer (offline.gif), or an image for a computer addressable with the SNMP protocol (SNMP.gif) is read in.

[0063] In the following steps S14 and S15, a hyperlink start variable ("SnmpHLStart") or a hyperlink end variable ("SnmpHLEnd") with the strings needed for generating a hyperlink is again documented via the command "userCompose". A hyperlink is an automated call of a datafile of the server by which a client sends the corresponding URL to the server comprising the datafile.

[0064] A program arrow from step S15 to step S2 that closes the program section between the steps S2 and S15 into a loop is entered in Figure 2 with broken lines. This program arrow indicates that this program section is multiply executed for

querying a plurality of computers reachable via the SNMP gateway. The queries of the individual computers are executed parallel. This is possible because the ZSNMP gateway is capable of multitasking.

[0065] In the following step S16, the strings representing the hyperlink are inserted into the datafile or HTML page at a predetermined location via the command "userPostReplaceString".

[0066] With the above-described program, an immediate query is performed to see whether one or more specific computers can be addressed with the SNMP gateway and whether a specific printing system is in stalled at one of the addressable computers. A corresponding image is then read in and a link pointing to the computer is generated and deposited in the HTML page. When the HTML page is not transmitted to the client, then the link can be either manually or automatically executed which starts a connection from the client that started the query directly to the computer with the queried printing system.

[0067] The client is thus switched from a server to a further server, where the first server sought a printing system for the client with which corresponding print orders can then be executed proceeding directly from the client. The search and switching procedure has been completely executed at the server; this shows that the necessary "intelligence" has to be provided only at the server and can be queried and operated by the client with a traditional browser.

[0068] The above program is only a highly simplified and abbreviated example for a query of reachable printing systems and automatic or semi-automatic switching to a desired printing system. This example is merely intended to indicate what possibilities are created by the inventive provision of an interpreter at the server.

[0069] Figure 3 is a flowchart showing how a print order can be processed with the inventive network. In step S18, the user inputs the print order and augments this with specific requests made of the quality of the printed product (colored, paper type, etc.) and necessary print features (number of copies, format, etc.).

[0070] In step S19, this print order is transmitted from the client 3, 4 to the server 1, 2. At the server, a database that contains the relevant data for connections

to printers or to further servers with connected printers or corresponding pre-processing and post-processing devices is updated S20. The updating S20 ensues in that one or more gateways are queried for corresponding printing systems. The data that are determined by this are entered in the local database deposited in the server. This updated database is evaluated in step S21 according to the parameters input by the user (print order, requests and prerequisites).

[0071] Subsequently, a check is carried out with a query as to whether a printer device suited for the print order has been found in the evaluation. When no suitable printer device has been found, the program execution branches to step S23 with which a message that the print order cannot be executed is sent to the client.

[0072] When, in contrast, the query in step S22 shows that a suitable printer device is present, a further query S24 checks whether the print order can be executed by the server. If the result of this query is no, then the client is switched S25 to a further server that can execute the print order, as was shown with reference to the program example from Figure 2. The print order together with the parameters (requests, prerequisites) is preferably directly communicated to the further server, where the parameters can by correspondingly modified as needed.

[0073] When, in contrast, the query S24 shows that the print order can be executed by the existing server, a check is carried out in a further query S26 to see whether the print order can be directly communicated to the printer device. When the printer device has a corresponding network coupling, then a direct link to the printer device can be produced, resulting in freeing up the server from the print order.

[0074] When the query from step S26 shows that a direct communication of the print order to the printer device is possible, then this is executed in step S27. When such a communication is not possible, then a link to the server or its gateway is set with the step S28 in order to communicate the print data to the printer device via the gateway in step S29. After the end of the printing event, a print confirmation S30 ensues from the server to the client, thus ending the method.

[0075] In this method, the print order is automatically forwarded to a suitable printer device. Who is available and suited for the processing of the print order is,

so to speak, negotiated between individual servers and printer devices. Such a method is also called "trading".

[0076] As presented above, the inventive network is particularly suited for the creation of a decentralized printing system that can comprise one or more print servers. Given an Internet application of the print server, each Internet client that has a corresponding access authorization to the server can use this for its print orders. The technical realization of such a decentralized printing system is extremely simple and merely requires the installation of an inventive interpreter in which the corresponding commands for the drive of the printer devices or the pre-processing and post-processing devices are installed. A plurality of servers can be provided with an inventive interpreter in a single network. This permits a print order to be handed off between a plurality of servers.

[0077] The invention, however, is not limited to a printing system but can be employed for the drive, monitoring, maintenance, etc., of arbitrary devices. Currently, there are considerable efforts to make household appliances network-capable. With an inventive server, they can be checked, queried and potentially placed into operation by a user via the Internet proceeding from an arbitrary location. Fundamentally, the administration and control of all technical devices is possible with the inventive network. In particular, it is suitable for the administration of data processing systems, telecommunication systems and switching systems. The inventive server is especially advantageous for super-regional systems since the inventive server can be driven proceeding from an arbitrary location of the network.

[0078] Another important advantage of the inventive network is that, due to the provision of an interpreter, it is not limited to a specific application; rather, due to the design possibility of a computer language, it is possible to generate the respective application with language elements similar to a standard language, resulting in the inventive system having maximum flexibility. There are already specialized program parts for most applications that merely have to be integrated into the interpreter. Such program parts are, for example, gateways, printer drivers, spoolers or other unique control programs. In particular, specialized communication protocols such as SNMP or DMI can be exploited for the local data transmission to the printer devices since these protocols -- in contrast to the HTTP protocol -- have a

significantly lower protocol overhead, shorter response times, a high performance, and allow a simple application. These specialized protocols or specific interfaces can be made accessible to a user who does not require extensive knowledge of this technology nor who needs to install corresponding, usually very complicated, software on his client in order to set up a communication to such specialized systems.

[0079] The inventive interpreter can be stored on a data carrier and be loaded into a server from it or via a network. The invention can be briefly summarized in the following way:

[0080] The present invention is directed to a client-server network, to an interpreter installable at the server of the network and to a method for operating such a client-server network.

[0081] The invention is characterized in that datafiles are deposited at the server, these datafiles being capable of being fetched by the client and inventively comprising both language elements executable at the client as well as language elements executable at the server. An interpreter is provided at the server, which interprets the language elements executable at the server and executes them.

[0082] According to a preferred embodiment of the invention, the language elements executable at the client correspond to a mark-up language such as, for example, SGML, XML, HTML, since the user, when establishing these datafiles, can use known aids for producing the datafiles -- these usually being standard text processing programs -- in order to provide individual applications at the server of the network that can be called by an arbitrary client with a traditional browser.

[0083] The invention is particularly suited for the control of devices, particularly of printers and printing systems and the corresponding pre-processing and post-processing devices, since the control intelligence is centrally deposited at the server and can thus be used by many clients, and the data transfer between the clients and the server is kept low.

[0084] Another aspect of the invention is that a trading of, for example, print orders between a plurality of servers can be realized with simple means.

[0085] The above-described methods and apparatus are illustrative of the principles of the present invention. Numerous modifications and adaptations will be readily apparent to those skilled in this art without departing from the spirit and scope of the present invention.

LIST OF REFERENCE CHARACTERS

1	first web server
2	second web server
3	first client
4	second client
5	data line (Intranet)
6	data line (Intranet)
7	data connection (Internet)
8	device
9	device
10	serial line
11	SNMP connection
12	gateway
13	device
14	line
15	communication program
16	WWW service program
17	storage
17a	free memory area
17b	locked memory area
18	interpreter

Method Steps

S1	initialization
S2	call and query of the SNMP gateway
S3	Windows?
S4	Windows printing system?
S5	storing
S6	UNIX?
S7	UNIX printing system?
S8	storing
S9	no SNMP?
S10	PING
S11	PING?
S12	storing
S13	UserCompose: image
S14	UserCompose: hyperlink start
S15	UserCompose: hyperlink end
S16	insertion of the hyperlink into HTML page
S17	end
S18	input of the print order

S19 transmission from client to server
S20 updating of a database
S21 evaluation of the database
S22 suitable printer device?
S23 message to client
S24 can server execute print order?
S25 switching to further server
S26 can communication be carried out to printer device?
S27 communication to printer device with direct link
S28 link to server
S29 communication via gateway
S30 print confirmation

ABSTRACT

The invention relates to a client-server network, to an interpreter that can be installed on the server of the network, and to a method for operating such a client-server network. Files callable by the client are stored on the server and which comprise both language elements that can be run on the client and on the server. An interpreter is provided on the server which interprets the language elements (e.g. SGML, XML, HTML), that can be run on the server and which runs the same. Known utilities can be used to create the files in order to provide individual applications on the server of the network by any client using a conventional browser. The invention is especially suited for controlling devices, in particular, printers/systems. The invention is additionally characterized in that a trading of, for example, print jobs between a plurality of servers can be implemented using simple means.

Appendix

```
<head>
<TITLE>SNMP Response</TITLE>
</HEAD>

<BODY>

<!--
 ------------------------------------------------------------------------
   Commentary: program start
 ------------------------------------------------------------------------
//-->

<!--
 ------------------------------------------------------------------------
   Commentary: function "STDefineString";  is called by inil
 ------------------------------------------------------------------------
//-->

<PSUB FUNC="STDefineString">
 <PTAG FUNC="userDefineString" VALUE="SNMPVALUE%i%" VALUE="Offline"></PTAG>
 <PTAG FUNC="userDefineString" VALUE="REPLACEMENT%i%" ></PTAG>
 <PTAG FUNC="userDefineString" VALUE="SnmpHLStart%i%"></PTAG>
 <PTAG FUNC="userDefineString" VALUE="SnmpHLEnd%i%"></PTAG>
 <PTAG FUNC="userDefineString" VALUE="Stdout%i%"></PTAG>
 <PTAG FUNC="userDefineString" VALUE="Stdin%i%"></PTAG>
 <PTAG FUNC="userDefineString" VALUE="Stderr%i%"></PTAG>
</PSUB>

<!--
 ------------------------------------------------------------------------
   Commentary: function "MTScanSystem"
 ------------------------------------------------------------------------
//-->

<PSUB FUNC="MTScanSystem">                          // Commentary: Section s2
 <PTAG FUNC="userSNMPGateway"
  VALUE="SNMPVALUE%i%"
  VALUE="1"
  VALUE="160.120.17.%i%"
  VALUE="public"
  VALUE=".1.3.6.1.2.1.1.1.0">
 </PTAG>

 <PBRANCH FUNC="userIf"                             //  Commentary: Section s3
  VALUE="SNMPVALUE%i%"
  VALUE="FNC>WIN"                                   // Commentary: Section s4
  VALUE=<PTAG FUNC="userSNMPGateway"                // Commentary: Section s5
              VALUE="SNMPVALUE%i%"
              VALUE="1"
              VALUE="160.120.17.%i%"
              VALUE="public"
              VALUE=".1.3.6.1.4.1.1552.92.2.0">
        </PTAG>
 </PBRANCH>

 <PBRANCH FUNC="userIf"                             // Commentary: Section s6
          VALUE="SNMPVALUE%i%"
          VALUE="FNC>SCO"                           // Commentary: section s7
          VALUE=<PTAG FUNC="userSNMPGateway"        //  Commentary: section s8
                      VALUE="SNMPVALUE%i%"
                      VALUE="1"
```

```
                    VALUE="160.120.17.%i%"
                    VALUE="public"
                    VALUE=".1.3.6.1.3.1.1.1.1.1.0">
                </PTAG>
</PBRANCH>

<PBRANCH FUNC="userIf"                                  // Commentary: Section s9
         VALUE="SNMPVALUE%i%"
         VALUE="FNC>Offline"
         VALUE=<PTAG FUNC="userSystem"                  // Commentary: Section s10
                     VALUE="Stdout%i%"
                     VALUE="Stdin%i%"
                     VALUE="Stderr%i%"
                     VALUE="0"
                     VALUE="5000"
                     VALUE="ping -n 1"
                     VALUE="160.120.17.%i%">
                </PTAG>
</PBRANCH>

<PBRANCH FUNC="userIf"
         VALUE="Stdout%i%"
         VALUE="FNC>Antwort"                            // Commentary: Section s11
         VALUE=<PTAG FUNC="userDefineString"            // Commentary: Section s12
                     VALUE="SNMPVALUE%i%"
                     VALUE="PING">
                </PTAG>
</PBRANCH>

<PTAG FUNC="userCompose"                                // Commentary: Section s13
      VALUE="SNMPVALUE%i%"
      VALUE="false"

      VALUE="REPLACEMENT%i%"
      VALUE="FNC>ISTREAM"
      VALUE="../images/imagestream.gif"

      VALUE="REPLACEMENT%i%"
      VALUE="FNC>PRISMA"
      VALUE="../images/prisma.gif"

      VALUE="REPLACEMENT%i%"
      VALUE="FCS>PING"
      VALUE="../images/ping.gif"

      VALUE="REPLACEMENT%i%"
      VALUE="FNC>Offline"
      VALUE="../images/offline.gif">
</PTAG>

                                                        // Commentary: Section s14
<PTAG FUNC="userCompose"
      VALUE="SNMPVALUE%i%"
     VALUE="false"

      VALUE="SnmpHLStart%i%"
      VALUE="FNC>ISTREAM"
      VALUE="<a HREF=\"http://www.ops.de\">"

      VALUE="SnmpHLStart%i%"
      VALUE="FNC>PRISMA"
      VALUE="<a HREF=\"http://160.120.17.%i%/pjm.html\" tar-
get=\"_blank\">"

      VALUE="SnmpHLStart%i%"
      VALUE="FNC>Offline"
      VALUE="">
</PTAG>
```

```
 <PTAG FUNC="userCompose"                                    // Commentary: Section s15
       VALUE="SNMPVALUE%i%"
       VALUE="false"

       VALUE="SnmpHLEnd%i%"
       VALUE="FNC>ISTREAM"
       VALUE="</a>"

       VALUE="SnmpHLEnd%i%"
       VALUE="FNC>PRISMA"
       VALUE="</a>"

       VALUE="SnmpHLEnd%i%"
       VALUE="FNC>Offline"
       VALUE="">
 </PTAG>
</PSUB>

<!--
 ------------------------------------------------------------------------
   Commentary: Function "STPrintSystem:  is called by s16
 ------------------------------------------------------------------------
//-->

<PSUB FUNC="STPrintSystem">
 <PTAG FUNC="userPreReplaceString" VALUE="SnmpHLStart%i%"></PTAG>
 <img src=<PTAG FUNC="userPreReplaceString" VALUE="REPLACEMENT%i%"></PTAG>
width="80" height="80"
      alt="160.120.17.%i%
<PTAG FUNC="userPreReplaceString" VALUE="SNMPVALUE%i%"></PTAG>">
 <PTAG FUNC="userPreReplaceString" VALUE="SnmpHLEnd%i%"></PTAG>
</PSUB>

<!--
  Kommentar: inil - s1
//-->

<PTAG FUNC="userSetSNMPRetries" VALUE="2"></PTAG>
<PTAG FUNC="userSetSNMPTimeout" VALUE="2000"></PTAG>

<PTAG FUNC="userFor"
      VALUE="i"
      VALUE="200"
      VALUE="255"
      VALUE="1"
      VALUE="0"
      VALUE="STDefineString">
</PTAG>

<!--
 ------------------------------------------------------------------------
   Commentary: Steps s2-s15 --> call of the corresponding
   sub-programs of sections s2-s15
 ------------------------------------------------------------------------
//-->

<PTAG FUNC="userFor"
      VALUE="i"
      VALUE="200"
      VALUE="255"
      VALUE="1"
      VALUE="-1"
      VALUE="MTScanSystem">
</PTAG>
```

```
<!--
 ------------------------------------------------------------------------
   Commentary: Step s16
 ------------------------------------------------------------------------
//-->

<p>
<PTAG FUNC="userFor"
      VALUE="i"
      VALUE="200"
      VALUE="255"
      VALUE="1"
      VALUE="0"
      VALUE="STPrintSystem">
</PTAG>
</p>

<!--
 ------------------------------------------------------------------------
   Commentary: Step s17 / end
 ------------------------------------------------------------------------
//-->


</BODY>
</HTML>
```

This redlined draft, generated by CompareRite (TM) - The Instant Redliner, shows the differences between -

original document : Q:\DOCUMENTS\YEAR 2001\P010402-HOFSTETTER-NETWORK INTERPRETOR METHOD\ORIGINAL SPECIFICATION.DOC

and revised document: Q:\DOCUMENTS\YEAR 2001\P010402-HOFSTETTER-NETWORK INTERPRETOR METHOD\COPY OF SUBSTITUTE SPECIFICATION.DOC

CompareRite found 376 change(s) in the text

Deletions appear as Overstrike text surrounded by []

Additions appear as Bold-Underline text

SPECIFICATION

TITLE

NETWORK, INTERPRETER FOR SUCH A NETWORK AND METHOD FOR OPERATING A NETWORK

BACKGROUND OF THE INVENTION

FIELD OF THE INVENTION

[0001] The invention is directed to a network, to an interpreter for such a network and to a method for operating a network. In particular, the invention is directed to a network for a printing system.

DESCRIPTION OF THE RELATED ART

[0002] As a rule, a ~~[plurality]~~ **number** of printers and printing systems and potentially corresponding pre-processing and post-processing devices that, for example, cut the printed paper to a specific format or bind it are connected to existing data networks such as, for example, LANs (local area networks) and WANs (wide area networks). The individual printers and printing systems differ greatly in ~~[terms of]~~ their performance capability; for example, ink jet printers with a print output of 4 pages per minute or high-performance printers with a print output of 40 pages or

more per minute can [~~thus~~] be connected to the data network. The printers and printing systems differ not only in terms of their printing output but also in terms of their printing quality. For example, **although** there are [~~thus~~] printers that print only in a single color (monochromatically), [~~in contrast whereto~~] color [~~printer~~] **printers** are also being increasingly utilized. Electrophotographic high-performance printer devices are currently in the position of printing two colors without further [~~ado, i.e. in what is referred to as a spot color mode~~] **effort, i.e., in a "spot color mode"** of highlight color mode. Such a printer, for example, is known by the name PAGESTREAM 200DSC of Océ Printing Systems GmbH.

[0003] The different printers and printing systems are usually also connected to the respective network with different system interfaces such as, for example, the SNMP (Simple Network Management Protocol) or the DMI (Desktop Management Interface). Although there is often a physical connection to a plurality of printers or[~~, respectively,~~] printing systems via the network, only a part of the printers can be addressed.

[0004] Given the rapid development of the Intranet and Internet, with which several LANs and WANs are connected to a single network[~~,~~] **(**particularly as a result of the introduction of the WWW service based on the HTTP protocol (Hypertext Transport Protocol)**)**, the number of printers and printing systems that can be fundamentally addressed is increasing explosively, [~~whereby the multiplicity~~] **with a corresponding increase in the number** of protocols for addressing the printers and printing systems [~~increases accordingly.~~

~~]~~.

[0005] This success of the Internet and of the Intranet was greatly promoted by the introduction of the World Wide Web (WWW) that can be simply operated by Internet users and allows the [~~most varied types~~] **widest variety** of information to be fetched from the greatest variety of different locations in the entire world. The WWW service of the Internet is based on the client-server principle. The communication ensues between a web server, [~~which is also referred to as WWW server and~~]**(or "WWW server")** which offers information, and a client that displays the information. The information are stored on the web server in pages, [~~whereby~~] **in which** the data in the pages are stored **in**, for example, [~~in what is referred to as HTML format~~] **the**

"HTML format" (Hypertext Markup Language). Other formats are also in use, such as~~[, for example,]~~ XML. These formats are derivatives of the underlying SGML format (Standard General Markup Language). These formats are referred to as markup languages since the make-up **of the language** can be described and defined with them.

[0006] A corresponding document is composed of normal text~~[, whereby]~~ **in which** control instructions~~[, what are referred to as "tags",]~~ **("tags")** are inserted into the text. Among other things, these tags influence the layout that is displayed later in the observation program, the browser, at the client. For example, there are ~~[thus]~~ tags to generate headings or tags that can modify the print image. The tags are always enclosed in **the bracketing** "< ... >".

[0007] The transmission of ~~[the]~~ information from the web server to the client ensues according to the HTTP protocol~~[, whereby the]~~**; this** information transmitted to the client is read by ~~[the browser]~~**its** installed ~~[thereat]~~ **browser** and the individual tags are interpreted~~[,]~~ so that the information ~~[are]~~ **is** displayed at the picture screen of the client in the predetermined fashion.

[0008] Since ~~[it is]~~ essentially only static images and ~~[texts]~~ **text** that can be displayed in the HTML format, the WWW service has been augmented by JAVA for mixing in multimedia elements, animations or programs. JAVA is an object-oriented programming language for web applications developed by SUN Microsystems. JAVA supports text, hypertext, graphics, audio and animation functions.

[0009] The program packets programmed in JAVA can either be loaded onto the client from the web server as ~~[what are referred to as JAVA applets]~~**"JAVA applets"** or can be pre-installed at the client as ~~[what are referred to as JAVA applications]~~**"JAVA applications"**. A modification of JAVA that is referred to as JAVAscript allows the insertion of program packets in the HTML page.

[0010] JAVA applications with which both administration functions at the printers and printing systems as well as the sending of print orders to the individual printers or~~[, respectively,]~~ printing systems can be implemented have been developed in order to drive printers and printing systems via the Internet. These JAVA programs represent significant progress over the previous situation since ~~[it makes]~~ **they make** it possible for the user of a client computer to drive a plurality of

printers and printing systems via the Internet ~~[independently]~~ **independent** of **the** platform **used**. The type of printing systems, i.e., the interface types, that can be ~~[drive]~~ **driven** by a client are determined by the respective JAVA application, ~~[whereby]~~ **in which** a separate program element is provided for each interface type. This means that ~~[the]~~**given the large variety of interfaces that are driven with such a JAVA application,** JAVA application must be all the more extensive ~~[the more different interfaces there are that are to be driven with]~~**. The same is true of the function scope that is driven given** such a JAVA application. ~~[The same is true of the function scope that is to be driven given such a JAVA application. This results therein that these JAVA applications]~~ **This produced JAVA applications that** are extensive programs with, for example, a data scope of **up to** 9 MB **or greater**. They thus make use of considerable computer power and memory space at the client and are therefore only expedient for users who wish to regularly drive different printers and printing systems.

[0011] It is also known to provide programs executable at the server that can be called by the client. These programs are deposited at the server either as completely compiled programs or as program code that can be interpreted by an interpreter. The programming language Perl is often employed for Internet applications for such an interpretable program code. It is especially suited for data bank applications to be executed at the server. However, pages that can be loaded from the server to the client can also be generated with Perl~~[, whereby]~~; Perl provides commands with which the make-up of corresponding pages can be constructed. The Perl program code can ~~[thereby]~~ also generate ~~[an]~~ HTML program code. When such a program is called at the server, the complete program code is interpreted and executed at the Server.

~~[The inventor of the present invention is unaware or Perl-based programs for driving printers, printing systems and corresponding pre-processing and post-processing devices.~~

~~]~~**[0012]** The publications of Jörn Heid, "Kettenreaktion" in iX 11/1998, pages 166-171, of Jörn Heid, "Was es sein darf" in iX 11/1998, pages 64-67 or of Rainer Klute, "Mehr als Applets" in iX 11/1998, pages 60-63 describe servlets that are Java programs stored in a server. The servlets can be linked in HTML pages and are

executed at the server when one of these pages is called. For example, parts of or complete HTML pages can be generated with such servlets. Servlets thus represent a possibility for linking Java programs executable at the server into HTML pages.

[0013] The publication of Jörn Heid, "Hand angelegt" in iX 11/1998, pages 68-70 discloses ~~[what are referred to as Java]~~**"Java** server ~~[pages]~~ **pages"** (JSP) ~~[wherein]~~ **in which** script code and HTML code can be contained in the same document, ~~[whereby]~~ **where** the scripts are replaced by their results before being sent to the client. This technique ~~[thus]~~ makes it possible to provide datafiles at the server that are called by the client and comprise language elements that can be executed both at the client as well as at the server. Similar techniques are known under the trademarks Live ~~[Wire]~~ **Wire™** and Active Server ~~[Pages.~~ **]Pages™.**

[0014] The publication by B. Merkleund, F. Pilhofer, "Perlin vor die Middleware" in iX 4/1999, pages 154-165 relates to the CORBA mapping for script languages. CORBA (Common Object Request Broken Architecture) is a complicated architecture for a query and communication service. All possible objects within a network can be communicated with CORBA~~[. It is therefore]~~**; thus it is** also possible to communicate objects provided on periphery devices with CORBA. To this end, however, it is necessary that these objects are fluent in a CORBA-specific protocol for communication. ~~[Further]~~ **Furthermore**, CORBA comprises an API (Application Program Interface) with which individually fabricated programs at the server can communicate with the CORBA system. These individual programs~~[, of course,]~~ can also be provided for the communication of periphery devices. It is also possible to map CORBA-independent protocols onto the CORBA protocol. ~~[The]~~ **This involves the** production of such a mapping or~~[, respectively,]~~ of independent programs ~~[is involved. The]~~**, as well as the** generation of CORBA objects at periphery devices ~~[is also involved since]~~**since communicating objects requires adherence to** the protocol prescribed by CORBA ~~[must be adhered to for the communication of objects, this being]~~**, which is** very extensive and complicated due to its universal applicability.

[0015] **German patent document** DE 197 04 694 A1 discloses a method and an apparatus for controlling a periphery device via the Internet that employs known

CGI scripts, which are programs deposited at the server independently of an HTML page but that can be called from HTML pages with a corresponding reference and can be executed at the server.

[0016] The Druckerbuch of Océ Printing Systems GmbH, ISBN 3-00-001019-X, 3rd Edition, October 1998, discloses a print server for professional, Internet-independent applications on pages 12-2 through 12-8. This print server is called PRISMApro7. Such a print server is composed of a high-performance personal computer with ~~[a]~~ corresponding software in order to be able to drive one or more printers, particularly a fast printer or high-performance printer. Such a print server is linked into a network and converts the incoming data streams into corresponding print data. ~~[Dependent]~~ **Depending** on the embodiment, the print server can process data in the formats AFDPS, Line/SF, PostScript, TIFF, PDF, LCDS, Line/O and PCL. Such print servers are especially utilized in data bank applications~~[, whereby]~~ **in which** a great quantity of data with variable data is printed from a data bank.

~~[Further]~~**[0017]** **Furthermore**, pages 14-2 through 14-12 in the Druckerbuch of Océ Printing Systems GmbH, ISBN 3-00-001019-X, 3rd Edition, October 1998, describes the print production control system Océ Domain7. In this system, one or more printers with pre-processing devices and post-processing devices are connected ~~[by means of]~~ **via** a network. ~~[The]~~ **This** controls and monitors the print production. ~~[It is]~~ **This system provides** a network-based client-server solution with a data bank that is installed on a central data bank server. All machine and operating data acquired in the production process are collected in this data bank and are offered to the various clients for later evaluations or can be exported into client-specific applications. Interfaces to the high-performance printers connected in the network and to the various hardware and software components are realized with Océ Domain7. The industry standards DMI (Desktop Management Interface), LMO (Large Mailroom Operations) and ODBC (Open Data Base Connectivity) are supported. Océ Domain7 is a high-performance control and monitoring system that, in particular, is employed in printing centers wherein high-performance printers are coupled to devices for pre-processing and for post-processing.

[0018] **International patent document** WO 99/18534 discloses a method with which queries from different servers can be processed in a computer network~~[, whereby]~~ **in which** the processing of the queries is automatically divided among the servers. To this end, the servers comprise ~~[what is referred to as a load]~~ **a "load** balance ~~[module]~~ **module"** with which the respective work load of the servers connected via the network is determined and the query is assigned to that server that exhibits the lowest work load. This method is particularly provided for employment in the Internet.

[0019] **European patent document** EP 0 874 306 A2 discloses a network printing system to which a plurality of clients can be connected, ~~[said]~~ **these** clients being capable of executing print orders at different printers ~~[by means of]~~ **via** print servers provided in the network. The print servers are fashioned with a layer architecture and comprise a communication interface. This communication interface can be provided with known program packets that automatically translate the print format of the print order into a print format suitable for the printer.

[0020] **European patent document** EP 0 872 792 A2 discloses a network for communication with printing systems that is based on the Internet. In particular, functions are provided ~~[here]~~ in order to correctly print out HTML datafiles that contain a reference to a further image datafile. To this end, an interpreter that executes the HTML datafile and can insert the corresponding image datafiles into the HTML datafiles is provided at the output device that is connected to the network. ~~[Languages]~~ **Languages/formats,** such as~~[, for example, JPIG [sic] and GF [sic]]~~ **JPEG and GIF,** that describe image datafiles are interpreted with these interpreters. ~~[On the other hand]~~ **Alternately**, it is also possible that a translation program for the translation of the image datafiles ~~[[...]]~~ **allows the provision of** a directly executable program code ~~[can be]~~ provided in the network instead of the interpreter.

[0021] **International patent document** WO 86/29663 discloses HTML datafiles into which executable scripts are linked. Such a script is executed at the web server and, for example, is programmed in an interpreter language such as Basic or Tool Controlling Language or in a compiler language such as~~[, for example,]~~ "C" and is compiled in a correspondingly runnable program. ~~[How]~~ **This document**

also discloses how such HTML datafiles can be produced and deposited at a server ~~[is also disclosed therein.~~

~~]~~.

SUMMARY OF THE INVENTION

[0022] The invention is based on the object of finding a simple technical solution for the drive of different printers and printing systems.

[0023] This object is achieved by a network ~~[having the features of claim 1,]~~ **for the interconnection of computers, comprising a client computer; a server computer that is configured to store datafiles and transmit them to the client computer when the client computer calls them by sending a corresponding datafile address to the server, wherein the datafiles are structured to contain both language elements executable at the client as well as language elements executable at the server; an interpreter in the server configured to interpret and execute the language elements executable at the server; a further logical or physical system comprising data of a different format than data exchanged between the server and the client; and a gateway installed at the server and integrated in the interpreter, the gateway being configured to set up a data connection to the further logical or physical system, and configured to automatically convert both incoming as well as outgoing data into appropriate data formats, the gateway configured to be called by language elements of the interpreter.**

[0024] **This object is also achived by** an interpreter for ~~[such]~~ a network ~~[having the features of claim 10 [sic] and]~~**, wherein the interpreter configured to be installed at a server of a network for interconnecting computers, the interpreter being configured to interpret and execute language elements executable at the server that are contained in a datafile stored at the server, wherein a client is configured to receive the datafile, the datafile comprising additional language elements executable at the client.**

[0025] **This object is also achieved** by a method for operating a network ~~[having the features of claim 16 [sic]. Advantageous developments of the invention are recited in the subclaims.~~

~~]~~**for the interconnection of computers having a server and a client, comprising storing datafiles on the server that are executable in the server and in the client; calling the datafiles by the client by sending a corresponding datafile address to the server; transmitting the datafiles by the server to the client in response to the calling the datafiles by the client; inquiring by the client to the server, which is a queried server, for a specific service offered by the server, the client using specific parameters of the service; determining by the queried server whether it can perform the inquired service; if the server can perform the service, the service performs the service by the server; if the server cannot perform the service, the server switches the client to a further server or device connected to the network that is capable of executing the service.**

[0026] The network of the invention is a network for the association of computers that connects at least one client to at least one server, ~~[whereby]~~ **in which** datafiles stored at the server can be called by the client by communicating a datafile address, ~~[as a result whereof]~~ **resulting in** a corresponding datafile ~~[is]~~ **being** transmitted to the server. These datafiles capable of being transmitted from the server to the client contain language elements that are executed at the client. An interpreter **is provided at the server** that can interpret and execute further language elements executable at the server that are contained in the datafiles stored at the server and fetchable by the client ~~[is provided at the server.~~

~~]~~.

[0027] With the invention, ~~[thus,]~~ datafiles are provided at the server that contain both language elements executable at the server as well as language elements executable at the client. ~~[Significant]~~ **This results in significant** advantages ~~[are achieved as a result thereof]~~, since it is no longer necessary -- as known, for example, for Perl programs executable at the server -- to generate language elements executable at the client ~~[by means of]~~ **via** language elements of a different language, which involves considerable programming outlay.

~~[Inventively, a gateway is integrated into an interpreter of a server, whereby language elements for calling the gateway are provided.~~

~~As a result thereof,]~~**[0028] Inventively, a gateway is integrated into an interpreter of a server in which language elements for calling the gateway are provided.**

This permits a computer language with which gateways can be directly called ~~[is]~~ **being** offered at the server. The integration of the gateway into the programming language allows the direct drive of arbitrary periphery devices from the programming language. As a result ~~[thereof]~~ **of this**, a user, who produces a datafile to be stored at the server, can handle the control of the periphery devices simultaneously with the production of this datafile. This is not possible in traditional systems since the corresponding interfaces can only be deposited at the server as CGI script, servlet, or the like by ~~[means]~~ **way** of programs that can be produced by specialists or must be mapped in an involved way at a known broker system.

[0029] Compared to the above-described CORBA system, the invention can be realized significantly more simply and with significantly less program code in the server and offers the user significantly more possibilities in the definition of his applications since it can be freely programmed as programming language.

[0030] An inventive network thus allows any user who has simple auxiliaries available for producing such datafiles stored at the server to also drive periphery devices that are in communication with the server either directly or via the network. In particular, printers can be driven in a simple way.

[0031] When the language elements executable at the client correspond to a mark-up language (such as, ~~[for example,]~~ SGML, HTML, XML), the datafiles can be edited with most standard editors and text processing programs. No knowledge of the mark-up language is often needed for producing simple mark-up datafiles since the corresponding syntax is automatically inserted by the editor or~~[, respectively,]~~ by the text processing program. Complex applications can thus be produced with the invention using text processing programs that are simple to use. The language elements executable at the server must merely be attached into the familiar datafiles fetchable from the server. The combination of language elements executable at the client as well as executable at the server in one datafile thus assures a significant facilitation for the user who establishes the datafiles at the server since he can produce these datafiles ~~[with means with which he is]~~ **using** familiar **mechanisms** and need not encode the commands executable at the client in another computer language upon whose execution they are artificially generated.

[0032] By providing an interpreter, which investigates the datafiles called by the client for language elements contained ~~[therein]~~ **within** and potentially interprets and executes these language elements at the server, the execution of an inherently arbitrary program at the server can be initiated by the transmission of only a single address from the client to the server. ~~[A]~~ **This creates the** possibility ~~[is thus created]~~ of being able to call arbitrary programs previously deposited in corresponding datafiles at the server with minimal programming outlay at the client and absolutely minimal data volume that must be transmitted between the server and the client. ~~[~~

~~]~~These programs can be used by a plurality of clients, so that a corresponding program packet need be deposited at only a single location in the network, **i.e.,** at the server.

[0033] The advantages achieved with the invention become especially clear when the data transmission in the network is based on a standardized data transmission such as~~[, for example,]~~ that according to the HTTP protocol that is applied in the Intranets and Internets. Given employment of such a standard, only a traditional browser without further, additional programs is required at the client in order to drive the corresponding functions. Up to now, involved programs, for example in JAVA, were loaded onto the clients~~[, as was initially explained, as a result whereof]~~ **resulting in** only the desired functions onto which the corresponding programs were installed or loaded ~~[can be]~~ **being** driven at the clients. Given the inventive network, in contrast, every client at which a browser is provided can call the corresponding programs at the server without additional programs having to be loaded onto the client. Only an access authorization to the respective server is required.

~~[A]~~**[0034]** **This provides a** critical functional advantage of the invention compared to the traditional system ~~[is comprised therein]~~ **in** that it is possible according to the invention that program elements that automatically switch the client to a further server are contained in the datafiles stored at the server. This switching ensues, for example, simply by transmitting the address of the further server from the original server to the client, this then being executed at the client, i.e., being sent to

further servers[, as a result whereof a] **resulting in a** connection [is] **being** set up between the further server and the client.

[0035] This switching function can also be designed as an automatic switching function[, whereby] **in which** a datafile is created at the server that contains the services of other servers, so that the client is automatically switched to another server that can provide the service when a client asks for a specific service at this one server and this server cannot provide the service. This functions appears to the client as though his order were being negotiated between the servers in order to identify [that] **a particular** server that can fill the order. This function of automatic forwarding is therefore also called "trading".

[0036] The trading also comprises the possibility that the server functions as a type of transit or[, respectively,] relay station[, whereby it]. **In this role, the server** switches a connection between the client and a further workstation such as[, for example,] a further server of a printing station or the like, [whereby] **where** the server is connected between the workstation and the client and correspondingly forwards the data. Since the server is provided with a gateway, the data transmission between the further workstation and the server and between the server and the client can be based on a different system or[, respectively,] different protocol, [whereby] **and where** the server correspondingly translates the data so that a friction-[free] **free/transparent** data transmission is possible.

[0037] When corresponding datafiles are deposited at the server, a printing system with such a client-server network allows every client who can access this server to address the printer devices drivable with the server and pre-processing and post-processing devices without [a] specific software [therefor] **for this** having to be installed at the client. The individual print orders can also be forwarded from a server to another server in order, for example, to be printed out by a printer that is especially suited for this specific print order. The server can be provided with gateways for the conversion of the data into specific print protocols or other types of transmission protocols as well, so that printers coupled to the network in the greatest variety of ways can be driven with a single server.

[0038] A print station [composer] **composed** of at least one printer with a potential pre-processing and post-processing device can be coupled to the network

in a simple way in that the devices of the print station (printer, pre-processing and post-processing devices) are connected to a server that is provided with an inventive interpreter and at which corresponding programs for the drive of the individual devices of the print station are stored in datafiles that can be fetched by the client. The devices of such a print station can then be driven by any arbitrary client via the network insofar as there is a corresponding access authorization to the server. The invention thus creates a possibility with which a network access to the print station that can be addressed with extremely simple technical ~~[means]~~ **mechanisms** merely by attaching a server to an existing print station.

[0039] The inventive network is fashioned for the drive of printers and corresponding pre-processing and post-processing devices. However, it can also be employed for the drive of any other devices since the basic principle of the present invention, providing an interpreter at a network server that interprets datafiles called by the client, can be transferred to further, arbitrary network applications.

DESCRIPTION OF THE DRAWINGS

[0040] The invention is explained in greater detail below by way of example with reference to the attached drawings. ~~[Shown therein are:]~~

Figure 1 **is** an inventive network in a block circuit diagram, shown schematically;

Figure 2 **is** an interface query in a flowchart that shows the program execution of the computer program attached as an appendix; and

Figure 3 **is** a trading method in the flowchart.

DETAILED DESCRIPTION OF THE INVENTION

[0041] Figure 1 shows an exemplary embodiment of an inventive network. The network comprises a first web server 1, a second web server 2 and a first and second client 3, 4. The first and second web server and the first client are connected to one another via data lines 5, 6 ~~[by means of]~~ **via** a LAN, ~~[whereby]~~ **where** the data line 5 connects the first web server to the first client and the data line 6 connects the first web server 1 to the second web server 2. An Intranet is operated on the LAN.

[0042] There is no fixed, physical data connection between the first web server 1 and the second client 4. A corresponding data connection 7~~[,]~~ **(**for example**,** via the telephone network~~[,]~~**)** is set up as needed, ~~[whereby]~~ **and** the Internet is employed as **a** transmission medium.

[0043] The first client 3 is, for example, an office computer of a user of the inventive network connected to the first web server 1 via the Intranet~~[,]~~**;** in contrast ~~[whereto]~~**,** the **second** client ~~[2]~~ **4** is a personal computer at the ~~[user=s]~~ **user's** home with which the user can dial in with a modem in the Internet and set up a connection to the first web server 1. Since the data connection 7 is not permanently present, it is shown with broken lines in Figure 1.

[0044] A respective browser is installed at both clients 3, 4, so that the two clients 3, 4 can communicate with the services known from the Internet with the web server 1. For example, these services ~~[are]~~ **include** Telnet **(**that enables a terminal simulation ~~[or FTP]~~**) and/or FTP (File Transfer Protocol**, with which datafiles can be transmitted**)**. The present exemplary embodiment employs the service of the World Wide Web (WWW) that ~~[has automatic recourse as needed to the]~~ **can automatically utilize** further services of the Internet such as, ~~[for example,]~~ FTP, News, Telnet, Gopher, E-mail, etc. Two devices 8, 9 are connected to the first web server 1, ~~[whereby]~~ **where** the device 8 is connected to the web server 1 ~~[by means of]~~ **via** a serial line (RS 232/V24) **10**, and a connection 11 according to the SNMP protocol (Simple Network Management Protocol) is installed between the device 9 and the server 1. The SNMP protocol is particularly employed for driving devices connected to a network. A gateway 12 is provided at the web server 1, ~~[said]~~ **this** gateway converting the data incoming via the SNMP connection 11 into the Internet protocol applied by the web server 1 on the data connections 5 through 7 or~~[, respectively,]~~ converting the data in the opposite direction according to the Internet protocol onto the SNMP protocol when they are transmitted from the web server 1 to the control device 9.

[0045] All hardware and software components at the web server that set up a connection to a communication system or~~[, respectively,]~~ network different from the Internet are a gateway in the sense of the invention. A plurality of gateways to other communication systems can be provided at the web server 1 that, for example, are

based on the DMI standard (Desktop Management Interface), the LP standard (Line Printer), SLP (Service Location Protocol), **and/or** the IPP standard (Internet Printing Protocol). Gateways to the PJMweb are possible. PJMweb is a web-based print client that has been developed by Océ. A gateway can also generate a terminal emulation. In particular, gateways can be provided that are fashioned for the drive of specific printer communication systems.

[0046] Since a connection to different networks can be produced via the gateways, an arbitrary plurality of devices can be fundamentally connected to a server provided with a gateway or~~[, respectively,]~~ can be driven by the server.

[0047] The second web server 2 is connected to a further device 13 via a line 14. To this end, a communication program 15 specifically fashioned for the device 13 is installed at the second server 2. This communication program 15 is a compiled program for ~~[what is referred to as a web-based management]~~**"web-based management"**. Such programs are respectively configured for a specific application such as~~[, for example, the drive of]~~ **driving** the device 13, ~~[as a result whereof]~~ **resulting in** only a single connection to the device 13 ~~[can be]~~ **being** set up and only a specific device type ~~[can be]~~ **being** driven with this program.

[0048] In addition to other service programs, the first and second web server 1, 2 comprise a WWW service program 16 that comprises the functions known from the Prior Art, i.e., that, ~~[given reception]~~ **upon receipt** of a URL, it reads the corresponding pages out from a data store 17 of the web server 1, 2 and sends them to the client 3, 4 that sent the URL. Inventively, an interpreter 18 is provided at the server 1, 2 with which the pages deposited in the store 17 are interpreted before being sent to a client 3, 4, i.e., that language elements contained in the pages are executed by the interpreter 18. The individual pages are stored in the HTML format, i.e. ~~[that]~~, they comprise standardized language elements that can be executed by the browser of the client 3, 4. The language elements executable by the interpreter 18 are independent of language elements executable by the browser. They can also be viewed as a supplement to the HTML format, for which reason the format of the stored pages can also be referred to as ~~[expanded HTML]~~**"expanded HTML"** format.

[0049] A program code of an exemplary program explained in greater detail below is ~~[recited]~~ **illustrated** in the appendix. ~~[First, the scope]~~ **A brief description** of the commands ~~[shall be briefly explained]~~ **used follows, although the specific use of these specific commands is exemplary**. Like most computer languages, the interpreter comprises commands for string processing, system commands and structure commands. As needed, the interpreter can be provided with further groups of commands. In addition to the ~~[aforementioned]~~ **previously mentioned** groups of commands, the interpreter comprises a command with which new commands of the interpreter can be generated. This command **is known** as "addfunction" with which an arbitrary command (function) can be constructed from arbitrary commands available on the server (machine language, operating systems or other standard language). "addfunction" is not employed in ~~[the]~~ HTML pages, but can be called by the user in a program development environment.

[0050] Important string processing commands are "userDefineString", "userGetCGIString", "userPreReplaceString", "userPostReplaceString" and "userCompose". String variables can be defined with "userDefineString". A string input at the client can be read with "userGetCGIString". The commands "userPreReplaceString" and "userPostReplaceString" serve for arranging strings at a predetermined location of the page, ~~[whereby]~~ **where** the string is arranged at the corresponding location immediately after the execution of the command given the command "userPreReplaceString"~~[, in contrast whereto]~~. **This contrasts with where** the string is only arranged at the predetermined location after the processing of all language elements given "userPostReplaceString". The string represented by the command "userPreReplaceString" can thus still be modified by the further language elements to be executed by the interpreter, in contrast ~~[whereto]~~ **to where** the string arranged with the command "userPostReplaceString" is placed at the end, ~~[as a result whereof a modification of the string is no longer possible.~~

]resulting in an unmodifiable string.

[0051] One of the most powerful system commands is "userSystem (...)", ~~[whereby]~~ **with which** a command of the operating system of the server or of a further program installed on the server can be ~~[input into the parentheses]~~ **provided as a parameter**. As a result ~~[thereof]~~, commands and programs established at the

server can be called with the interpreter. This command is called from an HTML page.

[0052] A sub-group of system commands are control commands that, for example, serve for the drive of a specific printer. These commands usually correspond to the respective printer system commands that are respectively converted as **a** command of the interpreter and can be supplemented by further, higher control commands.

[0053] The structure commands serve for producing a program structure with branchings, loops and the like. Corresponding structure commands are, for example, "userFor", "userGoSub" or "userIf". [

]Corresponding to the syntax of HTML, the commands executable by the interpreter 18 are also placed in **brackets** "<...>".

[0054] Given the commands of the interpreter 18, a distinction is made between two classes, namely one with commands that cannot be directly called by the client and a further class whose commands can be directly called by the client. Of the commands recited above by way of example, those that begin with "user" cannot be called by the client. This division into two classes of commands serves for security since, if a user of an arbitrary client could generate a "userSystem" command on the server, he could manipulate the server at will. In order to avoid such a manipulation, the store 17 is subdivided into a free memory area 17a and a locked memory area 17b. In the free memory area, the client can call pages and these are potentially supplied with parameters~~[,];~~ in contrast ~~[whereto],~~ the locked memory area 17b contains pages that do not receive parameters directly from the client or that cannot be called by the client. These pages are only called by the interpreter.

[0055] The interpreter ~~[thereby]~~ monitors the parameter handover and the program execution. The interpreter 18 is fashioned such that security-relevant commands are only executed when they are stored in the locked memory area 17b. A user at one of the clients can merely read the content of the free memory area 17a and directly address it. The pages deposited in the locked memory area 17b can only be addressed indirectly via the interpreter. [

]The administrator of the server can deposit control programs needed for the drive of the devices 8, 9 and 13 in the locked memory area 17b of the server, these control programs usually being security-relevant.

[0056] The program code ~~[recited]~~ **listed** in the appendix is explained in greater detail below, ~~[the program]~~ **and its** execution ~~[thereof being shown]~~ **is illustrated by the flowchart** in Figure 2 ~~[in a flowchart. An]~~**. In the** initialization ~~[is implemented in the]~~ step S1 ~~["ini, whereby]~~**"ini",** specific values and strings are set or~~[, respectively,]~~ defined for ~~[respectively]~~ **respective** querying servers.

[0057] The SNMP gateway 12 is called with ~~[the]~~ step S2, ~~[whereby]~~ **where** the value "1" indicates that this gateway is supposed to read something in, ~~[whereby]~~ **and** the result is stored in "SNMPVALUE". ~~[Dependent]~~ **Depending** on a user input, an address of a data bank entry ~~[(]~~**(a** managed object of an MIB) is ~~[thereby]~~ queried. The result of this query represents the computer type that is addressed via the SNMP interface.

~~[On the basis of]~~**[0058]** **Based on** the value stored in the variable "SNMPVALUE", a check is carried out in the next step S3 to see whether the operating system of the computer addressed via the SNMP gateway is **Microsoft** Windows. When the operating system is Windows, the program execution branches to the query S4 ~~[wherein]~~ **where** an SNMP value is read in anew and a check is carried out on the basis of this value to see whether a predetermined printing system ("Imagestream") runnable under windows is established at the computer. When the query S4 ~~[yields]~~ **indicates** the presence of such a printing system, this is stored in ~~[the]~~ step S5.

[0059] After the storing event in ~~[the]~~ step S5 or~~[, respectively,]~~ if one of the queries S3 or S4 was answered in the negative, the program execution changes to ~~[the]~~ step S6 with which a check is carried out to see whether the operating system of the computer addressed via the SNMP gateway is a ~~[Unix]~~ **UNIX** operating system. When the query ~~[yields]~~ **indicates** that the operating system is a ~~[Unix]~~ **UNIX** operating system, the program execution branches to ~~[the]~~ step S7 with which an SNMP is read in again, **and** a check ~~[being]~~ **is** carried out ~~[with reference thereto]~~ to see whether a ~~[Unix operating [sic]]~~ **UNIX printing** system (PJM or~~[, respectively,]~~

Prisma) is established on the computer. When the query yields such a printing system, then this is stored in ~~[the]~~ step S8.

[0060] After the storing event of step S8 or when one of the two queries S6 and S7 had a negative outcome, the program execution switches to a step S9 with which a check is carried out to see whether the computer addressed via the SNMP gateway is a computer that can be driven at all with the SNMP protocol. When the query ~~[yields]~~ **indicates** that the computer can fundamentally be driven with the SNMP protocol, then a PING query is called in step S10 and the result of the PING query is checked in step S11 to see whether the computer is operating (online). When this query S11 ~~[yields]~~ **indicates** that the computer is operating (online), then this is stored in step S12~~[,]~~; in contrast ~~[whereto]~~, when the computer is not operating (offline), then this corresponds to the default setting, so that no additional storing is necessary.

[0061] The PING query is a program module established at many servers that is not only present in the Internet. It is therefore called with the "userSystem" function already described above or is integrated into the interpreter as an independent command. A renewed programming of this program module is thus superfluous.

[0062] After the storing event of step S12 or when the query in step S9 ~~[yielded]~~ **indicated** that the computer is an SNMP computer or when the query in step S11 yielded that the computer was not operating, the program execution switches to ~~[the]~~ step S13 with which an image presenting the printing system is registered. This ensues with the command "userCompose" with which the variable "SNMPVALUE" documented by the above queries and storing events is evaluated (see the line under the command "userCompoase"), ~~[whereby]~~ **where** the result is "exclusive". This is prescribed in the following line by the value "false". This means that only a single image can be read into the variable "REPLACEMENT". Dependent on the above result, an image for ~~[a [...],]~~ an Imagestream printing system (imagestream.gif), a Prisma printing system (Prisma.gif) ~~[or]~~, an image for the non-operational status of the computer (offline.gif), or an image for a computer addressable with the SNMP protocol (SNMP.gif) is read in.

[0063] In the following steps S14 and S15, a hyperlink start variable ("SnmpHLStart") or~~[, respectively,]~~ a hyperlink end variable ("SnmpHLEnd") with the strings needed for generating a hyperlink is again documented ~~[by means of]~~ **via** the command "userCompose". A hyperlink is an automated call of a datafile of the server~~[, whereby]~~ **by which** a client sends the corresponding URL to the server comprising the datafile.

[0064] A program arrow from step S15 to step S2 that closes the program section between the steps S2 and S15 into a loop is entered in Figure 2 with broken lines. ~~[What is expressed with this]~~ **This** program arrow ~~[is]~~ **indicates** that this program section is multiply executed for querying a plurality of ~~[computer]~~ **computers** reachable via the SNMP gateway. The queries of the individual computers are executed parallel. This is possible because the ZSNMP gateway is capable of multitasking.

[0065] In the following step S16, the strings representing the hyperlink are inserted into the datafile or~~[, respectively,]~~ HTML page at a predetermined location ~~[by means of]~~ **via** the command "userPostReplaceString".

[0066] With the above-described program, ~~[thus,]~~ an immediate query is performed to see whether one or more specific computers can be addressed with the SNMP gateway and whether a specific printing system is in stalled at one of the addressable computers. A corresponding image is then read in and a link pointing to the computer is generated and deposited in the HTML page. When the HTML page is not transmitted to the client, then the link can be either manually or automatically executed~~[,]~~ **which starts** a connection from the client that started the query directly to the computer with the queried printing system ~~[being started therewith.~~

~~]~~.

[0067] The client ~~[was]~~ **is** thus switched from a server to a further server, ~~[whereby]~~ **where** the first server sought a printing system for the client with which corresponding print orders can then be executed proceeding directly from the client. The search and switching procedure has been completely executed at the server; this shows that the necessary "intelligence" has to be provided only at the server and can be queried and operated by the client with a traditional browser.

[0068] The above program is only a highly simplified and abbreviated example for a query of reachable printing systems and automatic or~~[, respectively,]~~ semi-automatic switching to a desired printing system. This example is merely intended to indicate what possibilities are created by the inventive provision of an interpreter at the server.

[0069] Figure 3 ~~[shows]~~ **is** a flowchart ~~[that shows]~~ **showing** how a print order can be processed with the inventive network. In ~~[a]~~ step S18, the user inputs the print order and augments this with specific requests made of the quality of the printed product (colored, paper type, etc.) and necessary print features (number of copies, format, etc.).

[0070] In step S19, this print order is transmitted from the client 3, 4 to the server 1, 2. At the server, a database that contains the relevant data for connections to printers or~~[, respectively,]~~ to further servers with connected printers or~~[, respectively,]~~ corresponding pre-processing and post-processing devices is updated S20. The updating S20 ensues in that one or more gateways are queried for corresponding printing systems. The data that are ~~[thereby]~~ determined **by this** are entered in the local database deposited in the server. ~~[~~

~~]~~This updated database is evaluated in step S21 according to the parameters input by the user (print order, requests and prerequisites).

[0071] Subsequently, a check is carried out with a query as to whether a printer device suited for the print order has been found in the evaluation. When no suitable printer device has been found, the program execution branches to ~~[the]~~ step S23 with which a message that the print order cannot be executed is sent to the client.

[0072] When, in contrast, the query in step S22 shows that a suitable printer device is present, a further query S24 checks whether the print order can be executed by the server. If the result of this query is no, then the client is switched S25 to a further server that can execute the print order, as was shown with reference to the program example from Figure 2. The print order together with the parameters (requests, prerequisites) is ~~[thereby]~~ preferably directly communicated to the further server, ~~[whereby]~~ **where** the parameters can by correspondingly modified as needed.

[0073] When, in contrast, the query S24 shows that the print order can be executed by the existing server, a check is carried out in a further query S26 to see whether the print order can be directly communicated to the printer device. When the printer device has a corresponding network coupling, then a direct link to the printer device can be produced, ~~[as a result whereof]~~ **resulting in freeing up** the server ~~[is no longer burdened by]~~ **from** the print order.

[0074] When the query from step S26 shows that a direct communication of the print order to the printer device is possible, then this is executed in step S27. ~~[~~ ~~]~~When such a communication is not possible, then a link to the server or~~[, respectively,]~~ its gateway is set with the step S28 in order to ~~[communication]~~ **communicate** the print data to the printer device via the gateway in step S29. After the end of the printing event, a print confirmation S30 ensues from the server to the client, **thus ending** the method ~~[being thus ended.~~

~~]~~.

[0075] In this method, the print order is automatically forwarded to a suitable printer device. Who is available and suited for the processing of the print order is, so to speak, negotiated between individual servers and printer devices. Such a method is ~~[therefore]~~ also called "trading".

[0076] As presented above, the inventive network is particularly suited for the creation of a decentralized printing system that can comprise one or more print servers. Given an Internet application of the print server, each Internet client that has a corresponding access authorization to the server can use this for its print orders. The technical realization of such a decentralized printing system is extremely simple and merely requires the installation of an inventive interpreter in which the corresponding commands for the drive of the printer devices or~~[, respectively,]~~ the pre-processing and post-processing devices are installed. A plurality of servers can be provided with an inventive interpreter in a single network. ~~[As a result thereof, it is also possible that]~~ **This permits** a print order ~~[is]~~ **to be** handed off between a plurality of servers.

[0077] The invention, however, is not limited to a printing system but can be employed for the drive, monitoring, maintenance, etc., of arbitrary devices.

Currently, ~~[thus,]~~ there are considerable efforts to make household appliances network-capable. With an inventive server, they can be checked, queried and potentially placed into operation by a user via the Internet proceeding from an arbitrary location. Fundamentally, the administration and control of all technical devices is possible with the inventive network. In particular, it is suitable for the administration of data processing systems, telecommunication systems and switching systems~~[, whereby it]~~**. The inventive server** is especially advantageous for super-regional systems since the inventive server can be driven proceeding from an arbitrary location of the network.

[0078] Another ~~[critical]~~ **important** advantage of the inventive network is that, due to the provision of an interpreter, it is not limited to a specific application; **rather,** due to the design possibility of a computer language, ~~[rather,]~~ it is possible to generate the respective application with language elements similar to a standard language~~[. As a result thereof,]~~**, resulting in** the inventive system ~~[has]~~ **having** maximum flexibility. There are already specialized program parts for most applications that merely have to be integrated into the interpreter. Such program parts are, for example, gateways, printer drivers, spoolers or other unique control programs. In particular, specialized communication protocols such as SNMP or DMI can be exploited for the local data transmission to the printer devices since these protocols -- in contrast to the HTTP protocol -- have a significantly lower protocol overhead, shorter response times ~~[and]~~, a high performance, and allow a simple application. These specialized protocols or~~[, respectively,]~~ specific interfaces can be made accessible to a user~~[, who need neither confront]~~ **who does not require extensive knowledge of** this technology nor **who needs to** install ~~[a]~~ corresponding, usually very complicated, software on his client in order to set up a communication to such specialized systems.

[0079] The inventive interpreter can be stored on a data carrier and be loaded into a server ~~[therefrom]~~ **from it** or via a network. ~~[~~

~~]~~The invention can be briefly summarized in the following way:

[0080] The present invention is directed to a client-server network, to an interpreter installable at the server of the network and to a method for ~~[the operation of]~~ **operating** such a client-server network.

[0081] The invention is characterized in that datafiles are deposited at the server, ~~[said]~~ **these** datafiles being capable of being fetched by the client and inventively comprising both language elements executable at the client as well as language elements executable at the server. An interpreter is provided at the server, ~~[this interpreting]~~ **which interprets** the language elements executable at the server and ~~[executing]~~ **executes** them.

[0082] According to a preferred embodiment of the invention, the language elements executable at the client correspond to a mark-up language such as, for example, SGML, XML, HTML, since the user, when establishing these datafiles, can use known aids for producing the datafiles -- these usually being standard text processing programs -- in order to provide individual applications at the server of the network that can be called by an arbitrary client with a traditional browser.

[0083] The invention is particularly suited for the control of devices, particularly of printers and printing systems and the corresponding pre-processing and post-processing devices, since the control intelligence is centrally deposited at the server and can thus be used by many clients, and the data transfer between the clients and the server is kept low.

[0084] Another aspect of the invention is that a trading of, for example, print orders between a plurality of servers can be realized with simple means.

[0085] The above-described methods and apparatus are illustrative of the principles of the present invention. Numerous modifications and adaptations will be readily apparent to those skilled in this art without departing from the spirit and scope of the present invention.

LIST OF REFERENCE CHARACTERS

1	first web server
2	second web server
3	first client
4	second client
5	data line (Intranet)
6	data line (Intranet)
7	data connection (Internet)
8	device
9	device
10	serial line

11 SNMP connection
12 gateway
13 device
14 line
15 communication program
16 WWW service program
17 storage
17a free memory area
17b locked memory area
18 interpreter

Method Steps
S1 initialization
S2 call and query of the SNMP gateway
S3 Windows?
S4 Windows printing system?
S5 storing
S6 UNIX?
S7 UNIX printing system?
S8 storing
S9 no SNMP?
S10 PING
S11 PING?
S12 storing
S13 UserCompose: image
S14 UserCompose: hyperlink start
S15 UserCompose: hyperlink end
S16 insertion of the hyperlink into HTML page
S17 end
S18 input of the print order
S19 transmission from client to server
S20 updating of a database
S21 evaluation of the database
S22 suitable printer device?
S23 message to client
S24 can server execute print order?
S25 switching to further server
S26 can communication be carried out to printer device?
S27 communication to printer device with direct link
S28 link to server
S29 commination via gateway
S30 print confirmation

BOX PCT
IN THE UNITED STATES DESIGNATED/ELECTED OFFICE
OF THE UNITED STATES PATENT AND TRADEMARK OFFICE
UNDER THE PATENT COOPERATION TREATY--CHAPTER II

REQUEST FOR APPROVAL OF DRAWING ADDITIONS

APPLICANT(S):	Andreas HOFSTETTER
ATTORNEY DOCKET NO.:	P01,0402
INTERNATIONAL APPLICATION NO:	PCT/EP00/04312
INTERNATIONAL FILING DATE:	12 May 2000
INVENTION:	NETWORK, INTERPRETER FOR SUCH A NETWORK, AND METHOD FOR OPERATING A NETWORK

Assistant Commissioner for Patents,
Washington, D.C. 20231

Sir:

Enclosed are three sheets of drawings showing in red, the addition of labels to Figure 1. Approval of the changes is respectfully requested.

Submitted by,

Mark Bergner (Reg. No. 45,877)

Mark Bergner
SCHIFF HARDIN & WAITE
PATENT DEPARTMENT
6600 Sears Tower
Chicago, Illinois 60606-6473
(312) 258-5779
Attorney for Applicant(s)

CUSTOMER NUMBER 26574

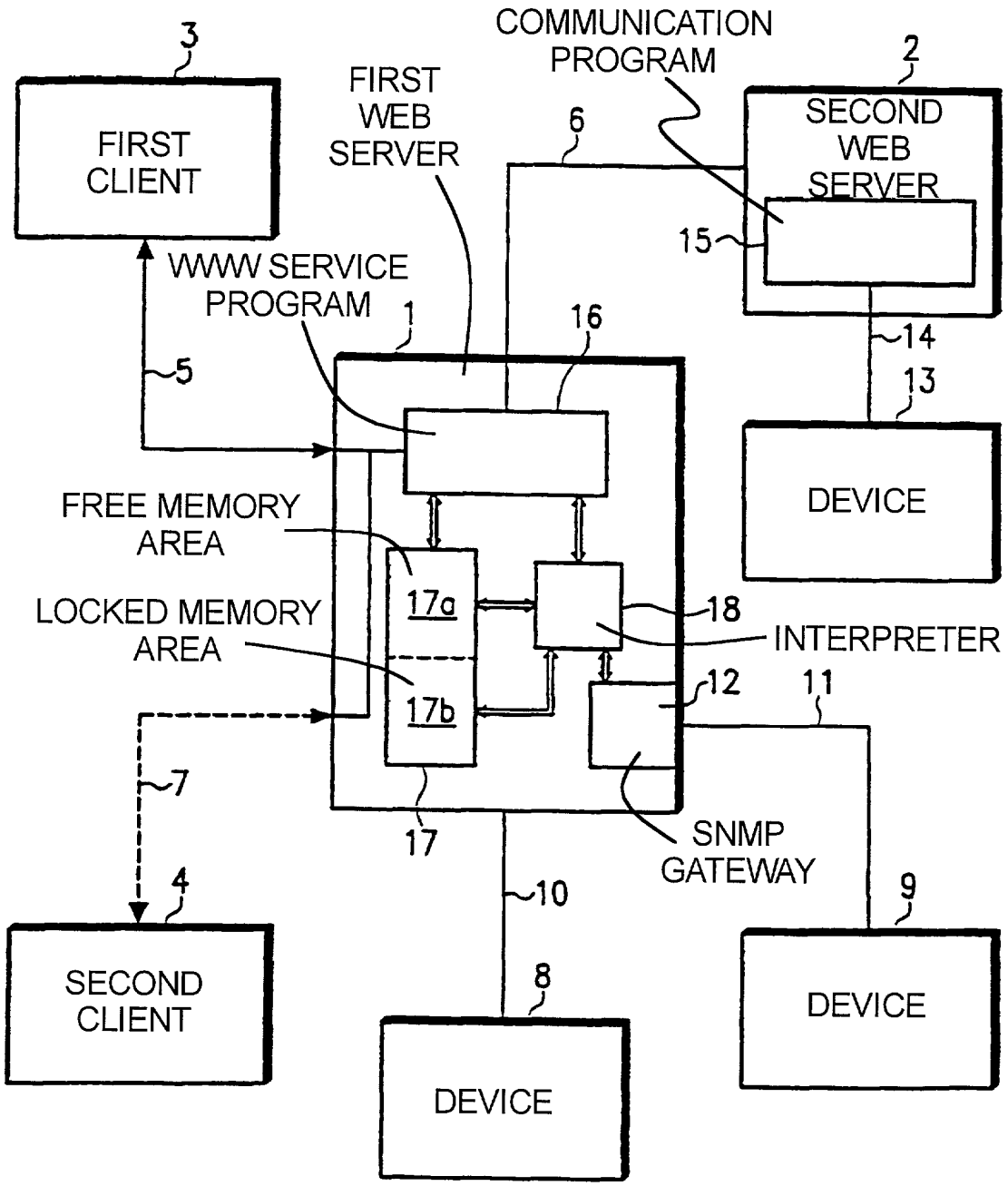
3
FIRST CLIENT
FIRST WEB SERVER
COMMUNICATION PROGRAM
2
SECOND WEB SERVER
6
15
WWW SERVICE PROGRAM
1
16
14
5
13
DEVICE
FREE MEMORY AREA
17a
18
INTERPRETER
LOCKED MEMORY AREA
17b
12
11
7
17
SNMP GATEWAY
4
10
9
SECOND CLIENT
8
DEVICE
DEVICE

FIG. 1

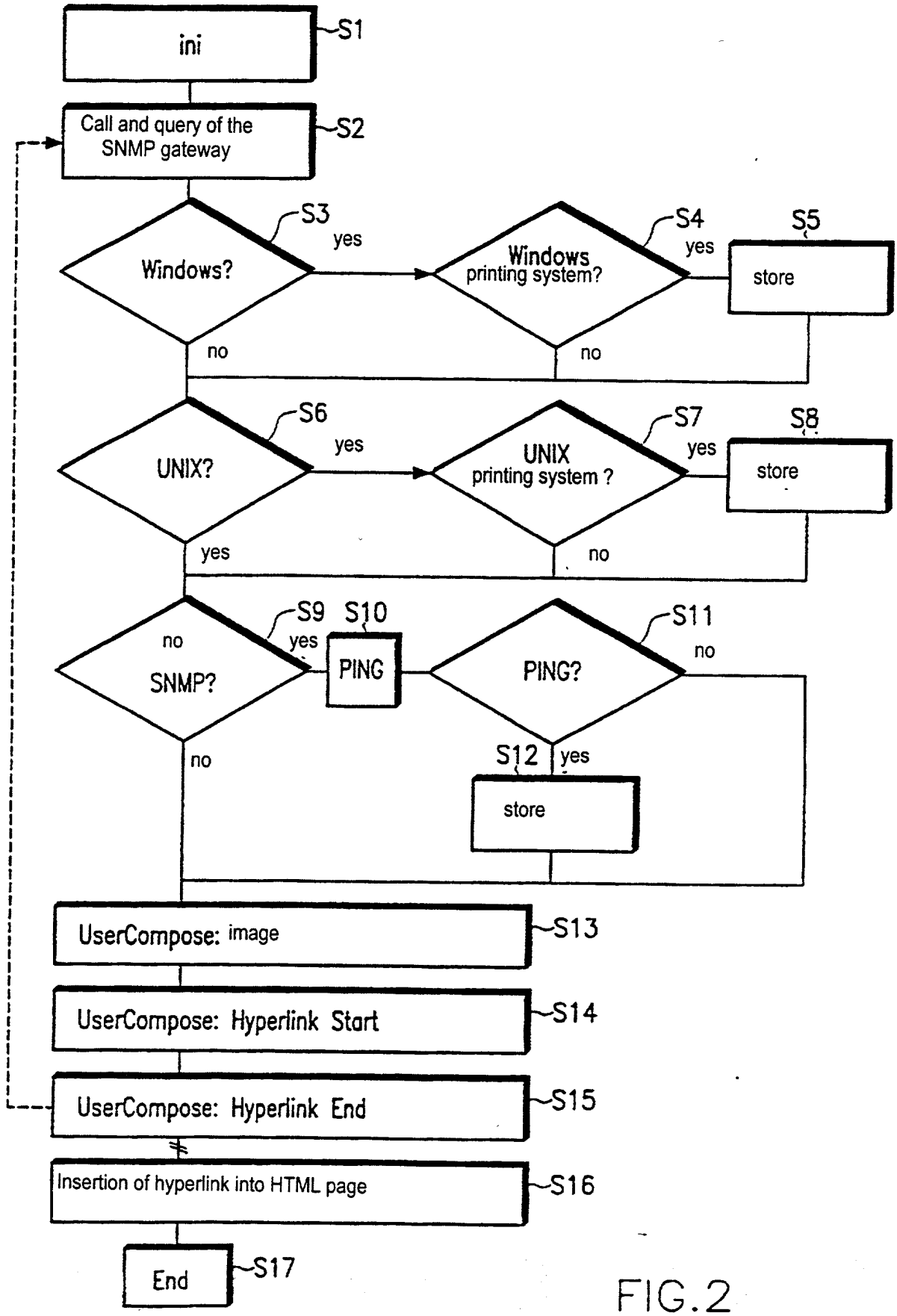
ini
S1
Call and query of the SNMP gateway
S2
Windows?
S3
yes
no
Windows printing system?
S4
yes
no
S5
store
UNIX?
S6
yes
yes
UNIX printing system ?
S7
yes
no
S8
store
SNMP?
no
S9
yes
S10
PING
PING?
S11
no
S12
yes
store
no
UserCompose: image
S13
UserCompose: Hyperlink Start
S14
UserCompose: Hyperlink End
S15
Insertion of hyperlink into HTML page
S16
End
S17

FIG.2

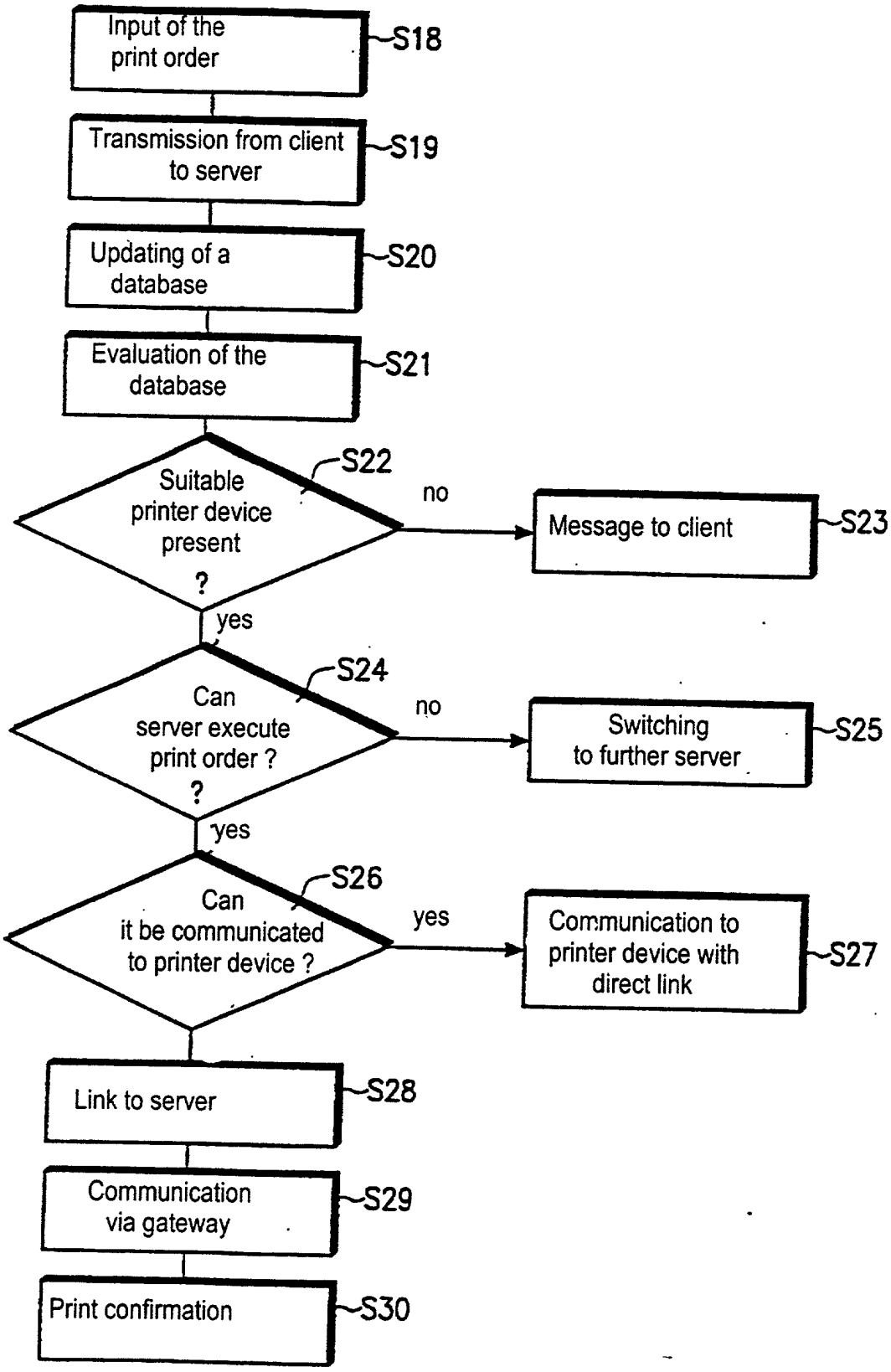
Input of the print order
S18
Transmission from client to server
S19
Updating of a database
S20
Evaluation of the database
S21
Suitable printer device present ?
S22
no
Message to client
S23
yes
Can server execute print order ?
S24
no
Switching to further server
S25
yes
Can it be communicated to printer device ?
S26
yes
Communication to printer device with direct link
S27
Link to server
S28
Communication via gateway
S29
Print confirmation
S30

FIG.3

NETWORK, INTERPRETER FOR SUCH A NETWORK AND METHOD FOR OPERATING A NETWORK

The invention is directed to a network, to an interpreter for such a network and to a method for operating a network. In particular, the invention is directed to a network for a printing system.

As a rule, a plurality of printers and printing systems and potentially corresponding pre-processing and post-processing devices that, for example, cut the printed paper to a specific format or bind it are connected to existing data networks such as, for example, LANs (local area networks) and WANs (wide area networks). The individual printers and printing systems differ greatly in terms of their performance capability; for example, ink jet printers with a print output of 4 pages per minute or high-performance printers with a print output of 40 pages or more per minute can thus be connected to the data network. The printers and printing systems differ not only in terms of their printing output but also in terms of their printing quality. For example, there are thus printers that print only in a single color (monochromatically), in contrast whereto color printer are also being increasingly utilized. Electrophotographic high-performance printer devices are currently in the position of printing two colors without further ado, i.e. in what is referred to as a spot color mode of highlight color mode. Such a printer, for example, is known by the name PAGESTREAM 200DSC of Océ Printing Systems GmbH.

The different printers and printing systems are usually also connected to the respective network with different system interfaces such as, for example, the SNMP (**S**imple **N**etwork **M**anagement **P**rotocol) or the DMI (**D**esktop **M**anagement **I**nterface). Although there is often a physical connection to a plurality of printers or, respectively, printing systems via the network, only a part of the printers can be addressed.

Given the rapid development of the Intranet and Internet, with which several LANs and WANs are connected to a single network, particularly as a result of the introduction of the WWW service based on the HTTP protocol (**H**ypertext **T**ransport **P**rotocol), the number of printers and printing systems that can be fundamentally

addressed is increasing explosively, whereby the multiplicity of protocols for addressing the printers and printing systems increases accordingly.

This success of the Internet and of the Intranet was greatly promoted by the introduction of the World Wide Web (WWW) that can be simply operated by Internet users and allows the most varied types of information to be fetched from the greatest variety of different locations in the entire world. The WWW service of the Internet is based on the client-server principle. The communication ensues between a web server, which is also referred to as WWW server and which offers information, and a client that displays the information. The information are stored on the web server in pages, whereby the data in the pages are stored, for example, in what is referred to as HTML format (**H**ypertext **M**arkup **L**anguage). Other formats are also in use such as, for example, XML. These formats are derivatives of the underlying SGML format (**S**tandard **G**eneral **M**arkup **L**anguage). These formats are referred to as markup languages since the make-up can be described and defined with them.

A corresponding document is composed of normal text, whereby control instructions, what are referred to as "tags", are inserted into the text. Among other things, these tags influence the layout that is displayed later in the observation program, the browser, at the client. For example, there are thus tags to generate headings or tags that can modify the print image. The tags are always enclosed in "< ... >".

The transmission of the information from the web server to the client ensues according to the HTTP protocol, whereby the information transmitted to the client is read by the browser installed thereat and the individual tags are interpreted, so that the information are displayed at the picture screen of the client in the predetermined fashion.

Since it is essentially only static images and texts that can be displayed in the HTML format, the WWW service has been augmented by JAVA for mixing in multimedia elements, animations or programs. JAVA is an object-oriented programming language for web applications developed by SUN Microsystems. JAVA supports text, hypertext, graphics, audio and animation functions.

The program packets programmed in JAVA can either be loaded onto the client from the web server as what are referred to as JAVA applets or can be pre-installed at the client as what are referred to as JAVA applications. A modification of JAVA that is referred to as JAVAscript allows the insertion of program packets in the HTML page.

JAVA applications with which both administration functions at the printers and printing systems as well as the sending of print orders to the individual printers or, respectively, printing systems can be implemented have been developed in order to drive printers and printing systems via the Internet. These JAVA programs represent significant progress over the previous situation since it makes it possible for the user of a client computer to drive a plurality of printers and printing systems via the Internet independently of platform. The type of printing systems, i.e. the interface types, that can be drive by a client are determined by the respective JAVA application, whereby a separate program element is provided for each interface type. This means that the JAVA application must be all the more extensive the more different interfaces there are that are to be driven with such a JAVA application. The same is true of the function scope that is to be driven given such a JAVA application. This results therein that these JAVA applications are extensive programs with, for example, a data scope of 9 MB. They thus make use of considerable computer power and memory space at the client and are therefore only expedient for users who wish to regularly drive different printers and printing systems.

It is also known to provide programs executable at the server that can be called by the client. These programs are deposited at the server either as completely compiled programs or as program code that can be interpreted by an interpreter. The programming language Perl is often employed for Internet applications for such an interpretable program code. It is especially suited for data bank applications to be executed at the server. However, pages that can be loaded from the server to the client can also be generated with Perl, whereby Perl provides commands with which the make-up of corresponding pages can be constructed. The Perl program code can thereby also generate an HTML program code. When such a program is called at the server, the complete program code is interpreted and executed at the Server. The

inventor of the present invention is unaware or Perl-based programs for driving printers, printing systems and corresponding pre-processing and post-processing devices.

The publications of Jörn Heid, "Kettenreaktion" in iX 11/1998, pages 166-171, of Jörn Heid, "Was es sein darf" in iX 11/1998, pages 64-67 or of Rainer Klute, "Mehr als Applets" in iX 11/1998, pages 60-63 describe servlets that are Java programs stored in a server. The servlets can be linked in HTML pages and are executed at the server when one of these pages is called. For example, parts of or complete HTML pages can be generated with such servlets. Servlets thus represent a possibility for linking Java programs executable at the server into HTML pages.

The publication of Jörn Heid, "Hand angelegt" in iX 11/1998, pages 68-70 discloses what are referred to as Java server pages (JSP) wherein script code and HTML code can be contained in the same document, whereby the scripts are replaced by their results before being sent to the client. This technique thus makes it possible to provide datafiles at the server that are called by the client and comprise language elements that can be executed both at the client as well as at the server. Similar techniques are known under the trademarks Live Wire and Active Server Pages.

The publication by B. Merkleund, F. Pilhofer, "Perlin vor die Middleware" in iX 4/1999, pages 154-165 relates to the CORBA mapping for script languages. CORBA (Common Object Request Broken Architecture) is a complicated architecture for a query and communication service. All possible objects within a network can be communicated with CORBA. It is therefore also possible to communicate objects provided on periphery devices with CORBA. To this end, however, it is necessary that these objects are fluent in a CORBA-specific protocol for communication. Further, CORBA comprises an API (Application Program Interface) with which individually fabricated programs at the server can communicate with the CORBA system. These individual programs, of course, can also be provided for the communication of periphery devices. It is also possible to map CORBA-independent protocols onto the CORBA protocol. The production of such a mapping or, respectively, of independent programs is involved. The generation of CORBA objects at periphery devices is also involved since the protocol prescribed by CORBA

must be adhered to for the communication of objects, this being very extensive and complicated due to its universal applicability.

DE 197 04 694 A1 discloses a method and an apparatus for controlling a periphery device via the Internet that employs known CGI scripts, which are programs deposited at the server independently of an HTML page but that can be called from HTML pages with a corresponding reference and can be executed at the server.

The Druckerbuch of Océ Printing Systems GmbH, ISBN 3-00-001019-X, 3rd Edition, October 1998, discloses a print server for professional, Internet-independent applications on pages 12-2 through 12-8. This print server is called PRISMApro®. Such a print server is composed of a high-performance personal computer with a corresponding software in order to be able to drive one or more printers, particularly a fast printer or high-performance printer. Such a print server is linked into a network and converts the incoming data streams into corresponding print data. Dependent on the embodiment, the print server can process data in the formats AFDPS, Line/SF, PostScript, TIFF, PDF, LCDS, Line/O and PCL. Such print servers are especially utilized in data bank applications, whereby a great quantity of data with variable data is printed from a data bank.

Further, pages 14-2 through 14-12 in the Druckerbuch of Océ Printing Systems GmbH, ISBN 3-00-001019-X, 3rd Edition, October 1998, describes the print production control system Océ Domain®. In this system, one or more printers with pre-processing devices and post-processing devices are connected by means of a network. The controls and monitors the print production. It is a network-based client-server solution with a data bank that is installed on a central data bank server. All machine and operating data acquired in the production process are collected in this data bank and are offered to the various clients for later evaluations or can be exported into client-specific applications. Interfaces to the high-performance printers connected in the network and to the various hardware and software components are realized with Océ Domain®. The industry standards DMI (**D**esktop **M**anagement **I**nterface), LMO (**L**arge **M**ailroom **O**perations) and ODBC (**O**pen **D**ata **B**ase **C**onnectivity) are supported. Océ Domain® is a high-performance control and monitoring system that, in particular, is employed in printing centers wherein high-

performance printers are coupled to devices for pre-processing and for post-processing.

WO 99/18534 discloses a method with which queries from different servers can be processed in a computer network, whereby the processing of the queries is automatically divided among the servers. To this end, the servers comprise what is referred to as a load balance module with which the respective work load of the servers connected via the network is determined and the query is assigned to that server that exhibits the lowest work load. This method is particularly provided for employment in the Internet.

EP 0 874 306 A2 discloses a network printing system to which a plurality of clients can be connected, said clients being capable of executing print orders at different printers by means of print servers provided in the network. The print servers are fashioned with a layer architecture and comprise a communication interface. This communication interface can be provided with known program packets that automatically translate the print format of the print order into a print format suitable for the printer.

EP 0 872 792 A2 discloses a network for communication with printing systems that is based on the Internet. In particular, functions are provided here in order to correctly print out HTML datafiles that contain a reference to a further image datafile. To this end, an interpreter that executes the HTML datafile and can insert the corresponding image datafiles into the HTML datafiles is provided at the output device that is connected to the network. Languages such as, for example, JPIG [sic] and GF [sic] that describe image datafiles are interpreted with these interpreters. On the other hand, it is also possible that a translation program for the translation of the image datafiles [...] a directly executable program code can be provided in the network instead of the interpreter.

WO 86/29663 discloses HTML datafiles into which executable scripts are linked. Such a script is executed at the web server and, for example, is programmed in an interpreter language such as Basic or Tool Controlling Language or in a compiler language such as, for example, "C" and is compiled in a correspondingly

runnable program. How such HTML datafiles can be produced and deposited at a server is also disclosed therein.

The invention is based on the object of finding a simple technical solution for the drive of different printers and printing systems.

This object is achieved by a network having the features of claim 1, an interpreter for such a network having the features of claim 10 [sic] and by a method for operating a network having the features of claim 16 [sic]. Advantageous developments of the invention are recited in the subclaims.

The network of the invention is a network for the association of computers that connects at least one client to at least one server, whereby datafiles stored at the server can be called by the client by communicating a datafile address, as a result whereof a corresponding datafile is transmitted to the server. These datafiles capable of being transmitted from the server to the client contain language elements that are executed at the client. An interpreter that can interpret and execute further language elements executable at the server that are contained in the datafiles stored at the server and fetchable by the client is provided at the server.

With the invention, thus, datafiles are provided at the server that contain both language elements executable at the server as well as language elements executable at the client. Significant advantages are achieved ss a result thereof, since it is no longer necessary -- as known, for example, for Perl programs executable at the server -- to generate language elements executable at the client by means of language elements of a different language, which involves considerable programming outlay.

Inventively, a gateway is integrated into an interpreter of a server, whereby language elements for calling the gateway are provided.

As a result thereof, a computer language with which gateways can be directly called is offered at the server. The integration of the gateway into the programming language allows the direct drive of arbitrary periphery devices from the programming language. As a result thereof, a user, who produces a datafile to be stored at the server, can handle the control of the periphery devices simultaneously with the

production of this datafile. This is not possible in traditional systems since the corresponding interfaces can only be deposited at the server as CGI script, servlet or

the like by means of programs that can be produced by specialists or must be mapped in an involved way at a known broker system.

Compared to the above-described CORBA system, the invention can be realized significantly more simply and with significantly less program code in the server and offers the user significantly more possibilities in the definition of his applications since it can be freely programmed as programming language.

An inventive network thus allows any user who has simple auxiliaries available for producing such datafiles stored at the server to also drive periphery devices that are in communication with the server either directly or via the network. In particular, printers can be driven in a simple way.

When the language elements executable at the client correspond to a mark-up language (such as, for example, SGML, HTML, XML), the datafiles can be edited with most standard editors and text processing programs. No knowledge of the mark-up language is often needed for producing simple mark-up datafiles since the corresponding syntax is automatically inserted by the editor or, respectively, by the text processing program. Complex applications can thus be produced with the invention using text processing programs that are simple to use. The language elements executable at the server must merely be attached into the familiar datafiles fetchable from the server. The combination of language elements executable at the client as well as executable at the server in one datafile thus assures a significant facilitation for the user who establishes the datafiles at the server since he can produce these datafiles with means with which he is familiar and need not encode the commands executable at the client in another computer language upon whose execution they are artificially generated.

By providing an interpreter, which investigates the datafiles called by the client for language elements contained therein and potentially interprets and executes these language elements at the server, the execution of an inherently arbitrary program at the server can be initiated by the transmission of only a single address from the client to the server. A possibility is thus created of being able to call arbitrary programs previously deposited in corresponding datafiles at the server with minimal

programming outlay at the client and absolutely minimal data volume that must be transmitted between the server and the client.

These programs can be used by a plurality of clients, so that a corresponding program packet need be deposited at only a single location in the network, at the server.

The advantages achieved with the invention become especially clear when the data transmission in the network is based on a standardized data transmission such as, for example, that according to the HTTP protocol that is applied in the Intranets and Internets. Given employment of such a standard, only a traditional browser without further, additional programs is required at the client in order to drive the corresponding functions. Up to now, involved programs, for example in JAVA, were loaded onto the clients, as was initially explained, as a result whereof only the desired functions onto which the corresponding programs were installed or loaded can be driven at the clients. Given the inventive network, in contrast, every client at which a browser is provided can call the corresponding programs at the server without additional programs having to be loaded onto the client. Only an access authorization to the respective server is required.

A critical functional advantage of the invention compared to the traditional system is comprised therein that it is possible according to the invention that program elements that automatically switch the client to a further server are contained in the datafiles stored at the server. This switching ensues, for example, simply by transmitting the address of the further server from the original server to the client, this then being executed at the client, i.e. being sent to further servers, as a result whereof a connection is set up between the further server and the client.

This switching function can also be designed as an automatic switching function, whereby a datafile is created at the server that contains the services of other servers, so that the client is automatically switched to another server that can provide the service when a client asks for a specific service at this one server and this server cannot provide the service. This functions appears to the client as though his order were being negotiated between the servers in order to identify that server that can fill the order. This function of automatic forwarding is therefore also called "trading".

The trading also comprises the possibility that the server functions as a type of transit or, respectively, relay station, whereby it switches a connection between the client and a further workstation such as, for example, a further server of a printing station or the like, whereby the server is connected between the workstation and the client and correspondingly forwards the data. Since the server is provided with a gateway, the data transmission between the further workstation and the server and between the server and the client can be based on a different system or, respectively, different protocol, whereby the server correspondingly translates the data so that a friction-free data transmission is possible.

When corresponding datafiles are deposited at the server, a printing system with such a client-server network allows every client who can access this server to address the printer devices drivable with the server and pre-processing and post-processing devices without a specific software therefor having to be installed at the client. The individual print orders can also be forwarded from a server to another server in order, for example, to be printed out by a printer that is especially suited for this specific print order. The server can be provided with gateways for the conversion of the data into specific print protocols or other types of transmission protocols as well, so that printers coupled to the network in the greatest variety of ways can be driven with a single server.

A print station composer of at least one printer with a potential pre-processing and post-processing device can be coupled to the network in a simple way in that the devices of the print station (printer, pre-processing and post-processing devices) are connected to a server that is provided with an inventive interpreter and at which corresponding programs for the drive of the individual devices of the print station are stored in datafiles that can be fetched by the client. The devices of such a print station can then be driven by any arbitrary client via the network insofar as there is a corresponding access authorization to the server. The invention thus creates a possibility with which a network access to the print station that can be addressed with extremely simple technical means merely by attaching a server to an existing print station.

The inventive network is fashioned for the drive of printers and corresponding pre-processing and post-processing devices. However, it can also be employed for the drive of any other devices since the basic principle of the present invention, providing an interpreter at a network server that interprets datafiles called by the client, can be transferred to further, arbitrary network applications.

The invention is explained in greater detail below by way of example with reference to the attached drawings. Shown therein are:

Figure 1 an inventive network in a block circuit diagram, shown schematically;

Figure 2 an interface query in a flowchart that shows the program execution of the computer program attached as an appendix; and

Figure 3 a trading method in the flowchart.

Figure 1 shows an exemplary embodiment of an inventive network. The network comprises a first web server 1, a second web server 2 and a first and second client 3, 4. The first and second web server and the first client are connected to one another via data lines 5, 6 by means of a LAN, whereby the data line 5 connects the first web server to the first client and the data line 6 connects the first web server 1 to the second web server 2. An Intranet is operated on the LAN.

There is no fixed, physical data connection between the first web server 1 and the second client 4. A corresponding data connection 7, for example via the telephone network, is set up as needed, whereby the Internet is employed as transmission medium.

The first client 3 is, for example, an office computer of a user of the inventive network connected to the first web server 1 via the Intranet, in contrast whereto the client 2 is a personal computer at the user's home with which the user can dial in with a modem in the Internet and set up a connection to the first web server 1. Since the data connection 7 is not permanently present, it is shown with broken lines in Figure 1.

A respective browser is installed at both clients 3, 4, so that the two clients 3, 4 can communicate with the services known from the Internet with the web server 1. For example, these services are Telnet that enables a terminal simulation or FTP, with which datafiles can be transmitted. The present exemplary embodiment employs the

service of the World Wide Web (WWW) that has automatic recourse as needed to the further services of the Internet such as, for example, FTP, News, Telnet, Gopher, E-mail, etc. Two devices 8, 9 are connected to the first web server 1, whereby the device 8 is connected to the web server 1 by means of a serial line (RS 232/V24), and a connection 11 according to the SNMP protocol (**S**imple **N**etwork **M**anagement **P**rotocol) is installed between the device 9 and the server 1. The SNMP protocol is particularly employed for driving devices connected to a network. A gateway 12 is provided at the web server 1, said gateway converting the data incoming via the SNMP connection 11 into the Internet protocol applied by the web server 1 on the data connections 5 through 7 or, respectively, converting the data in the opposite direction according to the Internet protocol onto the SNMP protocol when they are transmitted from the web server 1 to the control device 9. All hardware and software components at the web server that set up a connection to a communication system or, respectively, network different from the Internet are a gateway in the sense of the invention. A plurality of gateways to other communication systems can be provided at the web server 1 that, for example, are based on the DMI standard (**D**esktop **M**anagement **I**nterface), the LP standard (**L**ine **P**rinter), SLP (**S**ervice **L**ocation **P**rotocol), the IPP standard (**I**nternet **P**rinting **P**rotocol). Gateways to the PJMweb are possible. PJMweb is a web-based print client that has been developed by Océ. A gateway can also generate a terminal emulation. In particular, gateways can be provided that are fashioned for the drive of specific printer communication systems.

Since a connection to different networks can be produced via the gateways, an arbitrary plurality of devices can be fundamentally connected to a server provided with a gateway or, respectively, can be driven by the server.

The second web server 2 is connected to a further device 13 via a line 14. To this end, a communication program 15 specifically fashioned for the device 13 is installed at the second server 2. This communication program 15 is a compiled program for what is referred to as a web-based management. Such programs are respectively configured for a specific application such as, for example, the drive of the device 13, as a result whereof only a single connection to the device 13 can be set up and only a specific device type can be driven with this program.

In addition to other service programs, the first and second web server 1, 2 comprise a WWW service program 16 that comprises the functions known from the Prior Art, i.e. that, given reception of a URL, it reads the corresponding pages out from a data store 17 of the web server 1, 2 and sends them to the client 3, 4 that sent the URL. Inventively, an interpreter 18 is provided at the server 1, 2 with which the pages deposited in the store 17 are interpreted before being sent to a client 3, 4, i.e. that language elements contained in the pages are executed by the interpreter 18. The individual pages are stored in the HTML format, i.e. that they comprise standardized language elements that can be executed by the browser of the client 3, 4. The language elements executable by the interpreter 18 are independent of language elements executable by the browser. They can also be viewed as a supplement to the HTML format, for which reason the format of the stored pages can also be referred to as expanded HTML format.

A program code of an exemplary program explained in greater detail below is recited in the appendix. First, the scope of the commands shall be briefly explained. Like most computer languages, the interpreter comprises commands for string processing, system commands and structure commands. As needed, the interpreter can be provided with further groups of commands. In addition to the aforementioned groups of commands, the interpreter comprises a command with which new commands of the interpreter can be generated. This command as "addfunction" with which an arbitrary command (function) can be constructed from arbitrary commands available on the server (machine language, operating systems or other standard language. "addfunction" is not employed in the HTML pages but can be called by the user in a program development environment.

Important string processing commands are "userDefineString", "userGetCGIString", "userPreReplaceString", "userPostReplaceString" and "userCompose". String variables can be defined with "userDefineString". A string input at the client can be read with "userGetCGIString". The commands "userPreReplaceString"and "userPostReplaceString" serve for arranging strings at a predetermined location of the page, whereby the string is arranged at the corresponding location immediately after the execution of the command given the

command "userPreReplaceString", in contrast whereto the string is only arranged at the predetermined location after the processing of all language elements given "userPostReplaceString". The string represented by the command "userPreReplaceString" can thus still be modified by the further language elements to be executed by the interpreter, in contrast whereto the string arranged with the command "userPostReplaceString" is placed at the end, as a result whereof a modification of the string is no longer possible.

One of the most powerful system commands is "userSystem (...)", whereby a command of the operating system of the server or of a further program installed on the server can be input into the parentheses. As a result thereof, commands and programs established at the server can be called with the interpreter. This command is called from an HTML page.

A sub-group of system commands are control commands that, for example, serve for the drive of a specific printer. These commands usually correspond to the respective printer system commands that are respectively converted as command of the interpreter and can be supplemented by further, higher control commands.

The structure commands serve for producing a program structure with branchings, loops and the like. Corresponding structure commands are, for example, "userFor", "userGoSub" or "userIf".

Corresponding to the syntax of HTML, the commands executable by the interpreter 18 are also placed in "<...>".

Given the commands of the interpreter 18, a distinction is made between two classes, namely one with commands that cannot be directly called by the client and a further class whose commands can be directly called by the client. Of the commands recited above by way of example, those that begin with "user" cannot be called by the client. This division into two classes of commands serves for security since, if a user of an arbitrary client could generate a "userSystem" command on the server, he could manipulate the server at will. In order to avoid such a manipulation, the store 17 is subdivided into a free memory area 17a and a locked memory area 17b. In the free memory area, the client can call pages and these are potentially supplied with parameters, in contrast whereto the locked memory area 17b contains pages that do

not receive parameters directly from the client or that cannot be called by the client. These pages are only called by the interpreter. The interpreter thereby monitors the parameter handover and the program execution. The interpreter 18 is fashioned such that security-relevant commands are only executed when they are stored in the locked memory area 17b. A user at one of the clients can merely read the content of the free memory area 17a and directly address it. The pages deposited in the locked memory area 17b can only be addressed indirectly via the interpreter.

The administrator of the server can deposit control programs needed for the drive of the devices 8, 9 and 13 in the locked memory area 17b of the server, these control programs usually being security-relevant.

The program code recited in the appendix is explained in greater detail below, the program execution thereof being shown in Figure 2 in a flowchart. An initialization is implemented in the step S1 "ini", whereby specific values and strings are set or, respectively, defined for respectively querying servers.

The SNMP gateway 12 is called with the step S2, whereby the value "1" indicates that this gateway is supposed to read something in, whereby the result is stored in "SNMPVALUE". Dependent on a user input, an address of a data bank entry (managed object of an MIB) is thereby queried. The result of this query represents the computer type that is addressed via the SNMP interface.

On the basis of the value stored in the variable "SNMPVALUE", a check is carried out in the next step S3 to see whether the operating system of the computer addressed via the SNMP gateway is Windows. When the operating system is Windows, the program execution branches to the query S4 wherein an SNMP value is read in anew and a check is carried out on the basis of this value to see whether a predetermined printing system ("Imagestream") runnable under windows is established at the computer. When the query S4 yields the presence of such a printing system, this is stored in the step S5.

After the storing event in the step S5 or, respectively, if one of the queries S3 or S4 was answered in the negative, the program execution changes to the step S6 with which a check is carried out to see whether the operating system of the computer addressed via the SNMP gateway is a Unix operating system. When the query yields

that the operating system is a Unix operating system, the program execution branches to the step S7 with which an SNMP is read in again, a check being carried out with reference thereto to see whether a Unix operating [sic] system (PJM or, respectively, Prisma) is established on the computer. When the query yields such a printing system, then this is stored in the step S8.

After the storing event of step S8 or when one of the two queries S6 and S7 had a negative outcome, the program execution switches to a step S9 with which a check is carried out to see whether the computer addressed via the SNMP gateway is a computer that can be driven at all with the SNMP protocol. When the query yields that the computer can fundamentally be driven with the SNMP protocol, then a PING query is called in step S10 and the result of the PING query is checked in step S11 to see whether the computer is operating (online). When this query S11 yields that the computer is operating (online), then this is stored in step S12, in contrast whereto, when the computer is not operating (offline), then this corresponds to the default setting, so that no additional storing is necessary.

The PING query is a program module established at many servers that is not only present in the Internet. It is therefore called with the "userSystem" function already described above or is integrated into the interpreter as an independent command. A renewed programming of this program module is thus superfluous.

After the storing event of step S12 or when the query in step S9 yielded that the computer is an SNMP computer or when the query in step S11 yielded that the computer was not operating, the program execution switches to the step S13 with which an image presenting the printing system is registered. This ensues with the command "userCompose" with which the variable "SNMPVALUE" documented by the above queries and storing events is evaluated (see the line under the command "userCompoase"), whereby the result is "exclusive". This is prescribed in the following line by the value "false". This means that only a single image can be read into the variable "REPLACEMENT". Dependent on the above result, an image for a [...], an Imagestream printing system (imagestream.gif), a Prisma printing system (Prisma.gif) or an image for the non-operational status of the computer (offline.gif) or an image for a computer addressable with the SNMP protocol (SNMP.gif) is read in.

In the following steps S14 and S15, a hyperlink start variable ("SnmpHLStart") or, respectively, a hyperlink end variable ("SnmpHLEnd") with the strings needed for generating a hyperlink is again documented by means of the command "userCompose". A hyperlink is an automated call of a datafile of the server, whereby a client sends the corresponding URL to the server comprising the datafile.

A program arrow from step S15 to step S2 that closes the program section between the steps S2 and S15 into a loop is entered in Figure 2 with broken lines. What is expressed with this program arrow is that this program section is multiply executed for querying a plurality of computer reachable via the SNMP gateway. The queries of the individual computers are executed parallel. This is possible because the ZSNMP gateway is capable of multitasking.

In the following step S16, the strings representing the hyperlink are inserted into the datafile or, respectively, HTML page at a predetermined location by means of the command "userPostReplaceString".

With the above-described program, thus, an immediate query is performed to see whether one or more specific computers can be addressed with the SNMP gateway and whether a specific printing system is in stalled at one of the addressable computers. A corresponding image is then read in and a link pointing to the computer is generated and deposited in the HTML page. When the HTML page is not transmitted to the client, then the link can be either manually or automatically executed, a connection from the client that started the query directly to the computer with the queried printing system being started therewith.

The client was thus switched from a server to a further server, whereby the first server sought a printing system for the client with which corresponding print orders can then be executed proceeding directly from the client. The search and switching procedure has been completely executed at the server; this shows that the necessary "intelligence" has to be provided only at the server and can be queried and operated by the client with a traditional browser.

The above program is only a highly simplified and abbreviated example for a query of reachable printing systems and automatic or, respectively, semi-automatic

switching to a desired printing system. This example is merely intended to indicate what possibilities are created by the inventive provision of an interpreter at the server.

Figure 3 shows a flowchart that shows how a print order can be processed with the inventive network. In a step S18, the user inputs the print order and augments this with specific requests made of the quality of the printed product (colored, paper type, etc.) and necessary print features (number of copies, format, etc.).

In step S19, this print order is transmitted from the client 3, 4 to the server 1, 2. At the server, a database that contains the relevant data for connections to printers or, respectively, to further servers with connected printers or, respectively, corresponding pre-processing and post-processing devices is updated S20. The updating S20 ensues in that one or more gateways are queried for corresponding printing systems. The data that are thereby determined are entered in the local database deposited in the server.

This updated database is evaluated in step S21 according to the parameters input by the user (print order, requests and prerequisites).

Subsequently, a check is carried out with a query as to whether a printer device suited for the print order has been found in the evaluation. When no suitable printer device has been found, the program execution branches to the step S23 with which a message that the print order cannot be executed is sent to the client.

When, in contrast, the query in step S22 shows that a suitable printer device is present, a further query S24 checks whether the print order can be executed by the server. If the result of this query is no, then the client is switched S25 to a further server that can execute the print order, as was shown with reference to the program example from Figure 2. The print order together with the parameters (requests, prerequisites) is thereby preferably directly communicated to the further server, whereby the parameters can by correspondingly modified as needed.

When, in contrast, the query S24 shows that the print order can be executed by the existing server, a check is carried out in a further query S26 to see whether the print order can be directly communicated to the printer device. When the printer device has a corresponding network coupling, then a direct link to the printer device

can be produced, as a result whereof the server is no longer burdened by the print order.

When the query from step S26 shows that a direct communication of the print order to the printer device is possible, then this is executed in step S27.

When such a communication is not possible, then a link to the server or, respectively, its gateway is set with the step S28 in order to communication the print data to the printer device via the gateway in step S29. After the end of the printing event, a print confirmation S30 ensues from the server to the client, the method being thus ended.

In this method, the print order is automatically forwarded to a suitable printer device. Who is available and suited for the processing of the print order is, so to speak, negotiated between individual servers and printer devices. Such a method is therefore also called "trading".

As presented above, the inventive network is particularly suited for the creation of a decentralized printing system that can comprise one or more print servers. Given an Internet application of the print server, each Internet client that has a corresponding access authorization to the server can use this for its print orders. The technical realization of such a decentralized printing system is extremely simple and merely requires the installation of an inventive interpreter in which the corresponding commands for the drive of the printer devices or, respectively, the pre-processing and post-processing devices are installed. A plurality of servers can be provided with an inventive interpreter in a single network. As a result thereof, it is also possible that a print order is handed off between a plurality of servers.

The invention, however, is not limited to a printing system but can be employed for the drive, monitoring, maintenance, etc., of arbitrary devices. Currently, thus, there are considerable efforts to make household appliances network-capable. With an inventive server, they can be checked, queried and potentially placed into operation by a user via the Internet proceeding from an arbitrary location. Fundamentally, the administration and control of all technical devices is possible with the inventive network. In particular, it is suitable for the administration of data processing systems, telecommunication systems and switching systems, whereby it is

especially advantageous for super-regional systems since the inventive server can be driven proceeding from an arbitrary location of the network.

Another critical advantage of the inventive network is that, due to the provision of an interpreter, it is not limited to a specific application; due to the design possibility of a computer language, rather, it is possible to generate the respective application with language elements similar to a standard language. As a result thereof, the inventive system has maximum flexibility. There are already specialized program parts for most applications that merely have to be integrated into the interpreter. Such program parts are, for example, gateways, printer drivers, spoolers or other unique control programs. In particular, specialized communication protocols such as SNMP or DMI can be exploited for the local data transmission to the printer devices since these protocols -- in contrast to the HTTP protocol -- have a significantly lower protocol overhead, shorter response times and a high performance and allow a simple application. These specialized protocols or, respectively, specific interfaces can be made accessible to a user, who need neither confront this technology nor install a corresponding, usually very complicated software on his client in order to set up a communication to such specialized systems.

The inventive interpreter can be stored on a data carrier and be loaded into a server therefrom or via a network.

The invention can be briefly summarized in the following way:

The present invention is directed to a client-server network, to an interpreter installable at the server of the network and to a method for the operation of such a client-server network.

The invention is characterized in that datafiles are deposited at the server, said datafiles being capable of being fetched by the client and inventively comprising both language elements executable at the client as well as language elements executable at the server. An interpreter is provided at the server, this interpreting the language elements executable at the server and executing them.

According to a preferred embodiment of the invention, the language elements executable at the client correspond to a mark-up language such as, for example, SGML, XML, HTML, since the user, when establishing these datafiles, can use

known aids for producing the datafiles -- these usually being standard text processing programs -- in order to provide individual applications at the server of the network that can be called by an arbitrary client with a traditional browser.

The invention is particularly suited for the control of devices, particularly of printers and printing systems and the corresponding pre-processing and post-processing devices, since the control intelligence is centrally deposited at the server and can thus be used by many clients, and the data transfer between the clients and the server is kept low.

Another aspect of the invention is that a trading of, for example, print orders between a plurality of servers can be realized with simple means.

Appendix

```
<head>
<TITLE>SNMP Response</TITLE>
</HEAD>

<BODY>

<!--
 ------------------------------------------------------------------------
   Commentary: program start
 ------------------------------------------------------------------------
//-->

<!--
 ------------------------------------------------------------------------
   Commentary: function "STDefineString":  is called by init
 ------------------------------------------------------------------------
//-->

<PSUB FUNC="STDefineString">
 <PTAG FUNC="userDefineString" VALUE="SNMPVALUE%i%" VALUE="Offline"></PTAG>
 <PTAG FUNC="userDefineString" VALUE="REPLACEMENT%i%" ></PTAG>
 <PTAG FUNC="userDefineString" VALUE="SnmpHLStart%i%"></PTAG>
 <PTAG FUNC="userDefineString" VALUE="SnmpHLEnd%i%"></PTAG>
 <PTAG FUNC="userDefineString" VALUE="Stdout%i%"></PTAG>
 <PTAG FUNC="userDefineString" VALUE="Stdin%i%"></PTAG>
 <PTAG FUNC="userDefineString" VALUE="Stderr%i%"></PTAG>
</PSUB>


<!--
 ------------------------------------------------------------------------
   Commentary: function "MTScanSystem"
 ------------------------------------------------------------------------
//-->

<PSUB FUNC="MTScanSystem">                          // Commentary: Section s2
 <PTAG FUNC="userSNMPGateway"
  VALUE="SNMPVALUE%i%"
  VALUE="1"
  VALUE="160.120.17.%i%"
  VALUE="public"
  VALUE=".1.3.6.1.2.1.1.1.0">
 </PTAG>

 <PBRANCH FUNC="userIf"                             //  Commentary: Section s3
  VALUE="SNMPVALUE%i%"
  VALUE="FNC>WIN"                                   // Commentary: Section s4
  VALUE=<PTAG FUNC="userSNMPGateway"                // Commentary: Section s5
              VALUE="SNMPVALUE%i%"
              VALUE="1"
              VALUE="160.120.17.%i%"
              VALUE="public"
              VALUE=".1.3.6.1.4.1.1552.92.2.0">
        </PTAG>
 </PBRANCH>


 <PBRANCH FUNC="userIf"                             // Commentary: Section s6
          VALUE="SNMPVALUE%i%"
          VALUE="FNC>SCO"                           // Commentary. section s7
          VALUE=<PTAG FUNC="userSNMPGateway"        //  Commentary: section s8
                      VALUE="SNMPVALUE%i%"
                      VALUE="1"
```

```
                    VALUE="160.120.17.%i%"
                    VALUE="public"
                    VALUE=".1.3.6.1.3.1.1.1.1.1.0">
                </PTAG>
</PBRANCH>

<PBRANCH FUNC="userIf"                              // Commentary: Section s9
         VALUE="SNMPVALUE%i%"
         VALUE="FNC>Offline"
         VALUE=<PTAG FUNC="userSystem"              // Commentary: Section s10
                    VALUE="Stdout%i%"
                    VALUE="Stdin%i%"
                    VALUE="Stderr%i%"
                    VALUE="0"
                    VALUE="5000"
                    VALUE="ping -n 1"
                    VALUE="160.120.17.%i%">
                </PTAG>
</PBRANCH>

<PBRANCH FUNC="userIf"
         VALUE="Stdout%i%"
         VALUE="FNC>Antwort"                        // Commentary: Section s11
         VALUE=<PTAG FUNC="userDefineString"        // Commentary: Section s12
                    VALUE="SNMPVALUE%i%"
                    VALUE="PING">
                </PTAG>
</PBRANCH>

<PTAG FUNC="userCompose"                            // Commentary: Section s13
      VALUE="SNMPVALUE%i%"
      VALUE="false"

      VALUE="REPLACEMENT%i%"
      VALUE="FNC>ISTREAM"
      VALUE="../images/imagestream.gif"

      VALUE="REPLACEMENT%i%"
      VALUE="FNC>PRISMA"
      VALUE="../images/prisma.gif"

      VALUE="REPLACEMENT%i%"
      VALUE="FCS>PING"
      VALUE="../images/ping.gif"

      VALUE="REPLACEMENT%i%"
      VALUE="FNC>Offline"
      VALUE="../images/offline.gif">
</PTAG>
                                                    // Commentary: Section s14
<PTAG FUNC="userCompose"
      VALUE="SNMPVALUE%i%"
     VALUE="false"

      VALUE="SnmpHLStart%i%"
      VALUE="FNC>ISTREAM"
      VALUE="<a HREF=\"http://www.ops.de\">"

      VALUE="SnmpHLStart%i%"
      VALUE="FNC>PRISMA"
      VALUE="<a HREF=\"http://160.120.17.%i%/pjm.html\" tar-
get=\"_blank\">"

      VALUE="SnmpHLStart%i%"
      VALUE="FNC>Offline"
      VALUE="">
</PTAG>
```

```
<PTAG FUNC="userCompose"                                    // Commentary: Section s15
      VALUE="SNMPVALUE%i%"
      VALUE="false"

      VALUE="SnmpHLEnd%i%"
      VALUE="FNC>ISTREAM"
      VALUE="</a>"

      VALUE="SnmpHLEnd%i%"
      VALUE="FNC>PRISMA"
      VALUE="</a>"

      VALUE="SnmpHLEnd%i%"
      VALUE="FNC>Offline"
      VALUE="">
</PTAG>
</PSUB>

<!--
 ------------------------------------------------------------------------
   Commentary Function "STPrintSystem. is called by s16
 ------------------------------------------------------------------------
//-->

<PSUB FUNC="STPrintSystem">
 <PTAG FUNC="userPreReplaceString" VALUE="SnmpHLStart%i%"></PTAG>
 <img src=<PTAG FUNC="userPreReplaceString" VALUE="REPLACEMENT%i%"></PTAG>
width="80" height="80"
      alt="160.120.17.%i%
<PTAG FUNC="userPreReplaceString" VALUE="SNMPVALUE%i%"></PTAG>">
 <PTAG FUNC="userPreReplaceString" VALUE="SnmpHLEnd%i%"></PTAG>
</PSUB>

<!--
  Kommentar: ini1 - s1
//-->

<PTAG FUNC="userSetSNMPRetries" VALUE="2"></PTAG>
<PTAG FUNC="userSetSNMPTimeout" VALUE="2000"></PTAG>

<PTAG FUNC="userFor"
      VALUE="i"
      VALUE="200"
      VALUE="255"
      VALUE="1"
      VALUE="0"
      VALUE="STDefineString">
</PTAG>

<!--
 ------------------------------------------------------------------------
   Commentary: Steps s2-s15 → call of the corresponding
   sub-programs of sections s2-s15
 ------------------------------------------------------------------------
//-->

<PTAG FUNC="userFor"
      VALUE="i"
      VALUE="200"
      VALUE="255"
      VALUE="1"
      VALUE="-1"
      VALUE="MTScanSystem">
</PTAG>
```

```
<!--
 ------------------------------------------------------------------------
   Commentary: Step s16
 ------------------------------------------------------------------------
//-->

<p>
<PTAG FUNC="userFor"
      VALUE="i"
      VALUE="200"
      VALUE="255"
      VALUE="1"
      VALUE="0"
      VALUE="STPrintSystem">
</PTAG>
</p>

<!--
 ------------------------------------------------------------------------
   Commentary: Step s17 / end
 ------------------------------------------------------------------------
//-->


</BODY>
</HTML>
```

List of Reference Characters

1	first web server
2	second web server
3	first client
4	second client
5	data line (Intranet)
6	data line (Intranet)
7	data connection (Internet)
8	device
9	device
10	serial line
11	SNMP connection
12	gateway
13	device
14	line
15	communication program
16	WWW service program
17	storage
17a	free memory area
17b	locked memory area
18	interpreter

Method Steps

	S1	initialization
	S2	call and query of the SNMP gateway
	S3	Windows?
5	S4	Windows printing system?
	S5	storing
	S6	UNIX?
	S7	UNIX printing system?
	S8	storing
10	S9	no SNMP?
	S10	PING
	S11	PING?
	S12	storing
	S13	UserCompose: image
15	S14	UserCompose: hyperlink start
	S15	UserCompose: hyperlink end
	S16	insertion of the hyperlink into HTML page
	S17	end
	S18	input of the print order
20	S19	transmission from client to server
	S20	updating of a database
	S21	evaluation of the database
	S22	suitable printer device?
	S23	message to client
25	S24	can server execute print order?
	S25	switching to further server
	S26	can communication be carried out to printer device?
	S27	communication to printer device with direct link
	S28	link to server
30	S29	commination via gateway
	S30	print confirmation

Claims

1. Network for the interconnection of computers, whereby at least one computer acts as server (1, 2) and one computer acts as client (3, 4), and datafiles stored at the server (1, 2) are transmitted from the server (1, 2) to the client (3, 4) when the client (3, 4) calls them by sending a corresponding datafile address to the server (1, 2), and
the datafiles contain both language elements executable at the client (3, 4) as well as language elements executable at the server (1, 2), and
an interpreter is present at the server (1, 2) for interpretation and execution of the language elements executable at the server, and
a gateway (12) is installed at the server (1, 2) that can set up a data connection to a further logical and/or physical system, whereby the data of the further system comprise a different format than the data exchanged between the server (1, 2) and the client (3, 4), and the gateway (12) automatically converts both the incoming as well as the outgoing data into the appropriate data formats, and
whereby the gateway (12) is integrated in the interpreter and can be called by language elements of the interpreter (18).

2. Network according to claim 1, whereby a plurality of gateways (12) are integrated in the interpreter and can be called by language elements of the interpreter (18).

3. Network according to claim 1 or claim 2, whereby the interpreter (18) is fashioned such at the server (1, 2) that the language elements executable at the server are executed at the server (3, 4) after the calling of the datafiles by a client and before the transmission of the datafiles to the client (3, 4).

4. Network according to one of the claims 1 through 3, whereby the datafile address corresponds to the URL format and the server (1, 2) is a web server, so that the datafiles can be called with an Internet browser installed at the client (3, 4).

5. Network according to one of the claims 1 through 4, whereby the datafiles stored at the server (1, 2) and fetchable by the client (3, 4) correspond to the format of a mark-up language that is expanded by the language elements executable at the server.

6. Network according to one of the claims 1 through 5, whereby a respective gateway (12) is provided for the conversion of the data in one or more of the following formats: SNMP, LP, PJMweb, ftp.

7. Network according to one of the claims 1 through 6, whereby the gateway or, respectively, gateways (12) are integrated in the interpreter and can be called by language elements of the interpreter (18).

8. Network according to one of the claims 1 through 7, whereby programs for the drive of at least one printer and/or pre-processing or post-processing devices are installed at the server (1, 2) and these programs can be called by the interpreter (18).

9. Interpreter for a network according to one of the preceding claims, that can be installed at a server (1, 2) of a network for the interconnection of computers and is fashioned for the interpretation and execution of language elements executable at the server (1, 2) that are contained in datafiles stored at the server (1, 2), whereby these datafiles by a client (3, 4) with the transmission of an address and contain additional language elements executable at the client (3, 4).

10. Interpreter according to claim 9, whereby the interpreter comprises a command for generating string entries in the datafile.

11. Interpreter according to claim 9 or 10, comprising a command for setting string entries at a predetermined location of the datafile.

12. Interpreter according to one of the claims 9 through 11, comprising a command for reading in a string transmitted from the client (3, 4) to the server (1, 2) and for storer [sic] the string into a predetermined variable.

13. Interpreter according to one of the claims 9 through 12, comprising a command for calling a gateway and querying a system connected to the gateway.

14. Interpreter according to one of the claims 9 through 13, whereby the interpreter comprises a group of client commands that can be called both proceeding from the client as well as from the server, and comprises a group of server commands that can only be called proceeding from the server.

15. Interpreter according to one of the claims 9 through 14, whereby the interpreter is stored on a data carrier.

16. Method for the operation of a network for the interconnection of computers that is fashioned according to one of the claims 1 through 8, whereby datafiles stored at servers (1, 2) are transmitted from one of the servers (1, 2) to a client (3, 4) when the client (3, 4) calls them by sending a corresponding datafile address to the server (1, 2) and the servers respectively offers [sic] the clients one or more services, whereby,
given a client inquiry for a specific service with specific parameters on which the service is based, the queried server determines whether it can perform the service and, when the server determines that it cannot perform the service, it switches a further server or to [sic] a device connected to the network that can execute the service to the client.

17. Method for the operation of a network according to claim 16, whereby one of the services offered by the servers is the execution of a print order, and
the server forwards the print order to a other server or directly to a printer device when the server itself cannot execute the print order.

18. method for the operation of a network according to claim 16 or 17, whereby the server comprises a data bank in which information about the services offered in the network are stored, so that, given a client query, a determination can be made on the basis of these data banks as to whether the desired service is present in the network.

19. Method for the operation of a network according to one of the claims 16 through 18, whereby the switching to a further server or to a device connected to the network is implemented by generating the address of the further server or of the device and by communicating the address to the querying client.

20. Method for the operation of a network according to one of the claims 16 through 19, comprising an interpreter according to one of the claims 9 through 15.

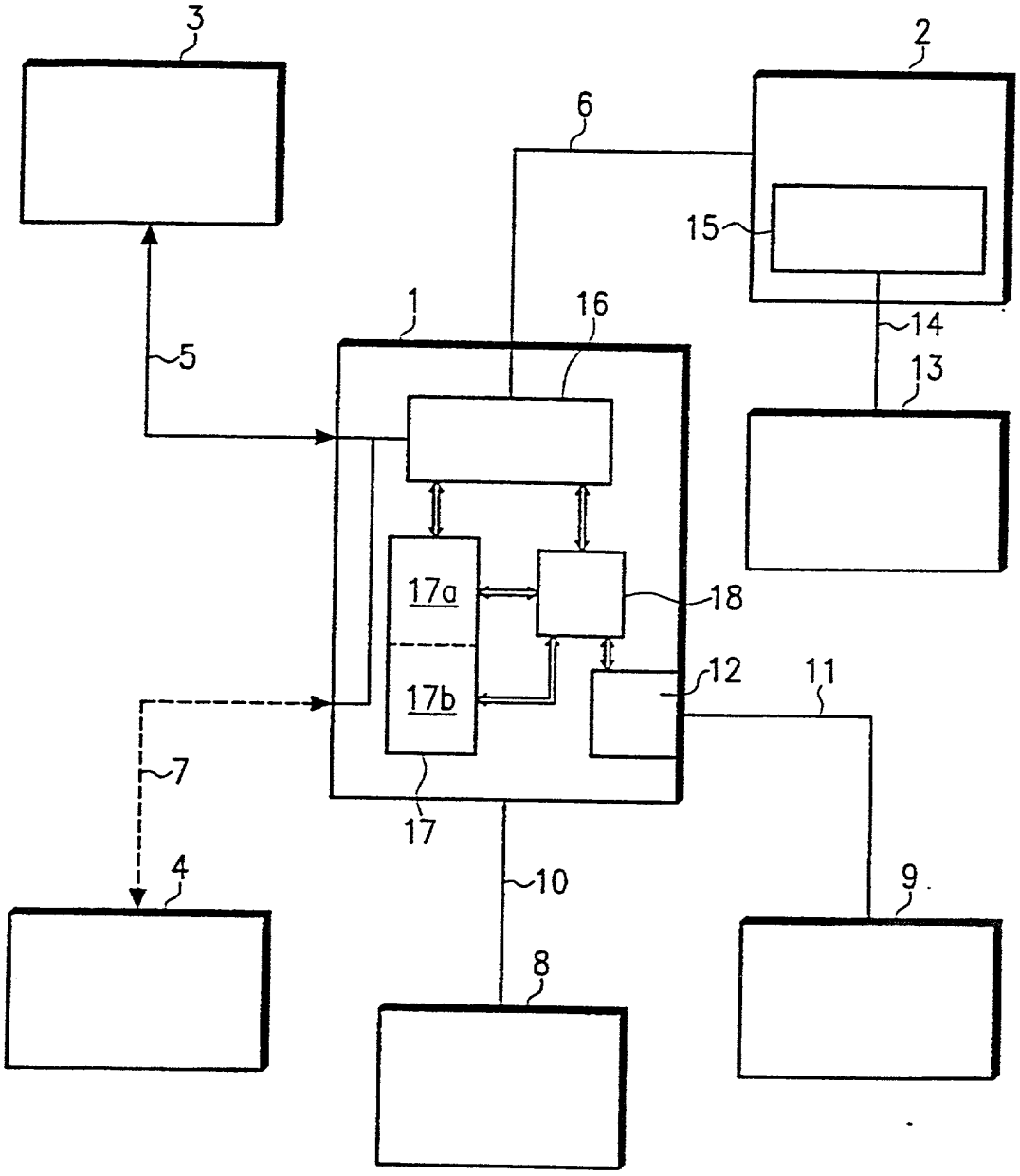

FIG. 1

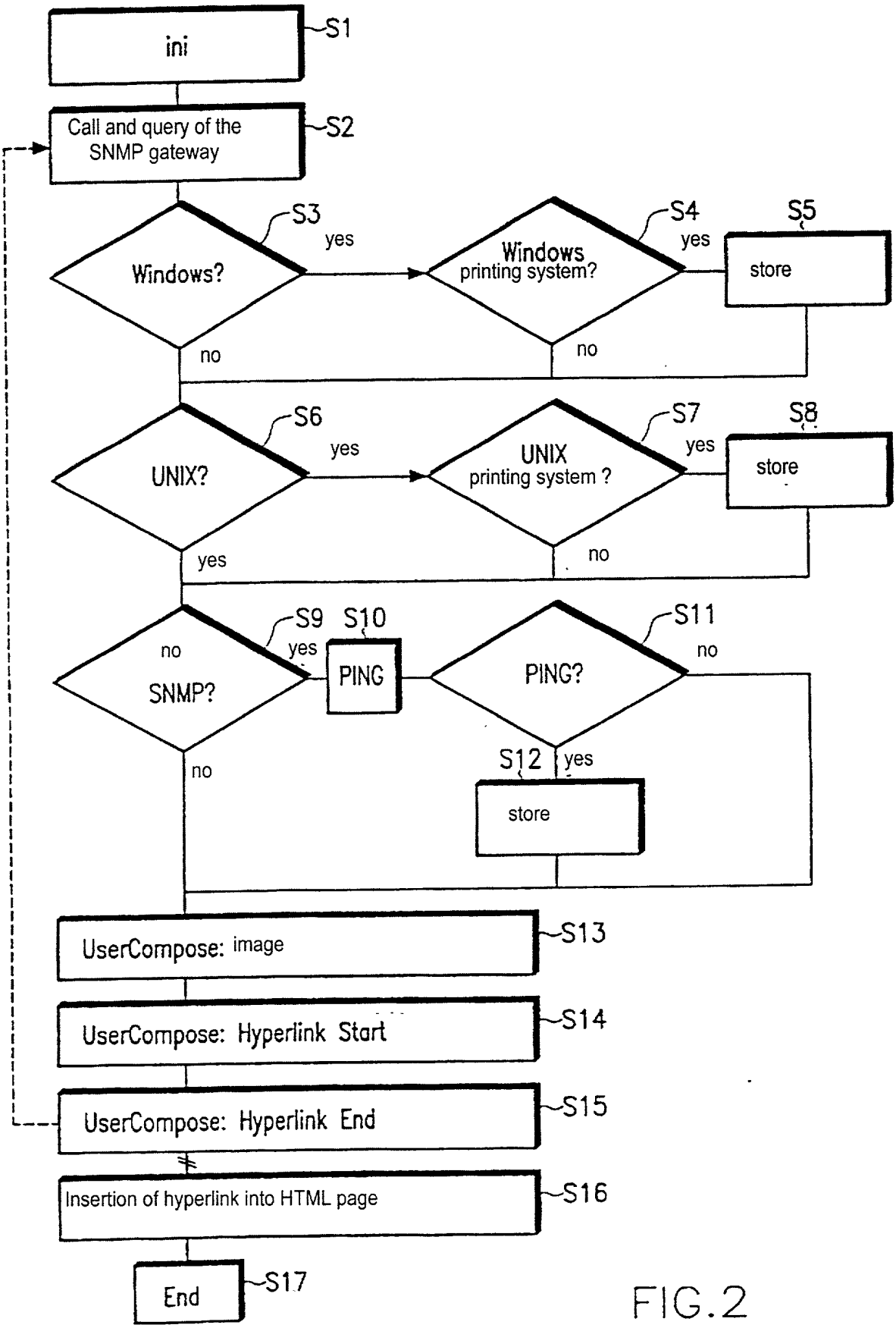
ini
S1
Call and query of the SNMP gateway
S2
Windows?
S3
yes
no
Windows printing system?
S4
yes
no
S5
store
UNIX?
S6
yes
yes
UNIX printing system ?
S7
yes
no
S8
store
SNMP?
no
S9
yes
S10
PING
PING?
S11
no
S12
yes
store
no
UserCompose: image
S13
UserCompose: Hyperlink Start
S14
UserCompose: Hyperlink End
S15
Insertion of hyperlink into HTML page
S16
End
S17

FIG.2

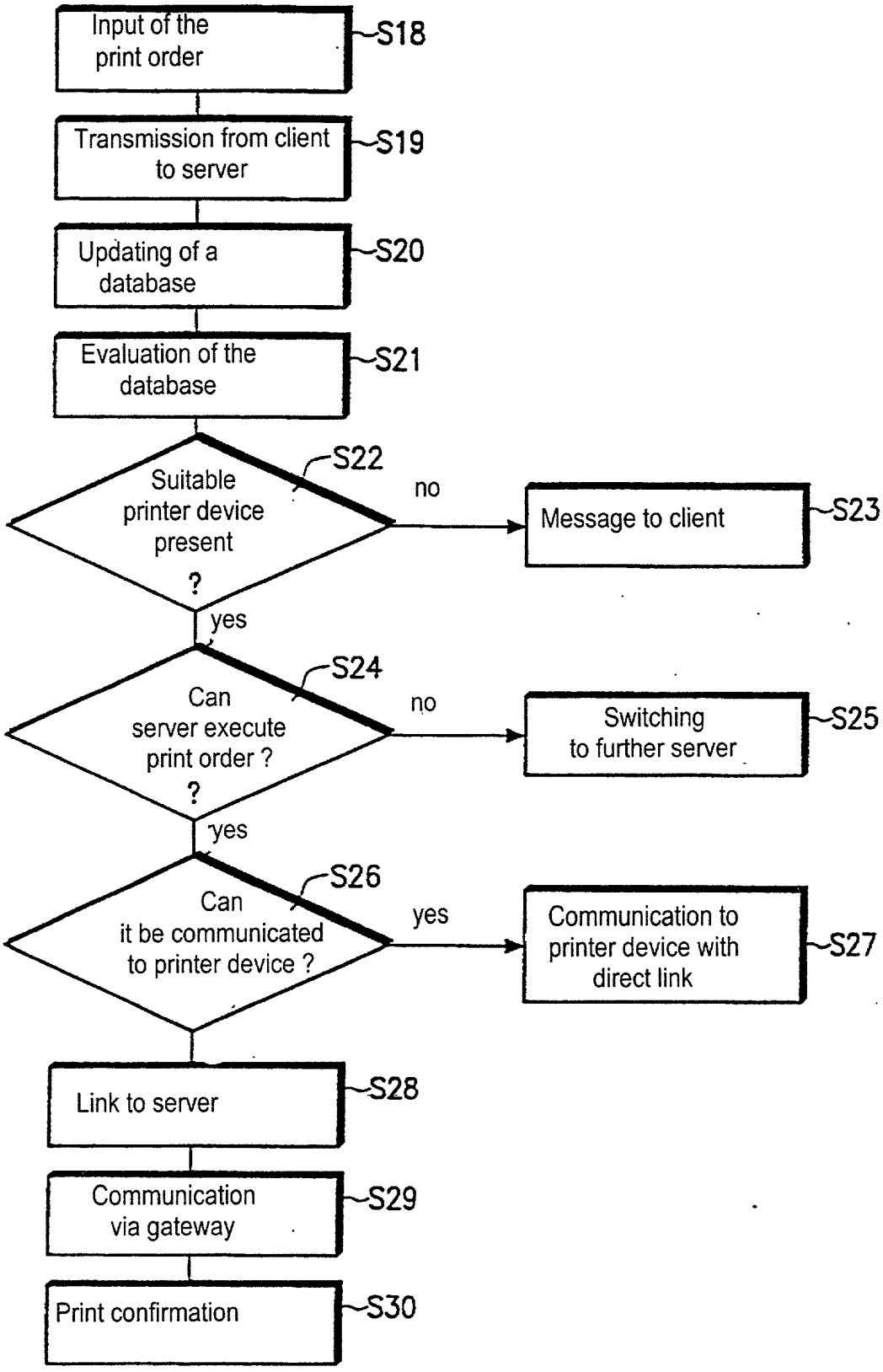
Input of the print order
S18
Transmission from client to server
S19
Updating of a database
S20
Evaluation of the database
S21
Suitable printer device present ?
S22
no
Message to client
S23
yes
Can server execute print order ?
S24
no
Switching to further server
S25
yes
Can it be communicated to printer device ?
S26
yes
Communication to printer device with direct link
S27
Link to server
S28
Communication via gateway
S29
Print confirmation
S30

FIG.3

DECLARATION AND POWER OF ATTORNEY FOR PATENT APPLICATION
ERKLÄRUNG FÜR PATENTANMELDUNGEN MIT VOLLMACHT
German Language Declaration

Als nachstehend benannter Erfinder erkläre ich hiermit an Eides Statt:

dass mein Wohnsitz, meine Postanschrift, und meine Staatsangehörigkeit den im Nachstehenden nach meinem Namen aufgeführten Angaben entsprechen,

dass ich, nach bestem Wissen der ursprüngliche, erste und alleinige Erfinder (falls nachstehend nur ein Name angegeben ist) oder ein ursprünglicher, erster und Miterfinder (falls nachstehend mehrere Namen aufgeführt sind) des Gegenstandes bin, für des dieser Antrag gestellt wird und für den ein Patent beantragt wird für die Erfindung mit dem Titel:

(zutreffendes ankreuzen)

☐ hier beigefügt ist.

☒ am_____ als
PCT internationale Anmeldung
PCT Anmeldungsnummer _____
eingereicht wurde und am _____
abgeändert wurde (falls tatsächlich abgeändert).

Ich bestätige hiermit, dass ich den Inhalt der obigen Patentanmeldung einschliesslich der Ansprüche durchgesehen und verstanden habe, die eventuell durch einen Zusatzantrag wie oben erwähnt abgeändert wurde.

Ich erkenne meine Pflicht zur Offenbarung irgendwelcher Informationen, die für die Prüfung der vorliegenden Anmeldung in Einklang mit Absatz 37, Bundesgesetzbuch, Paragraph 1.56 von Wichtigkeit sind, an.
Ich beanspruche hiermit ausländische Prioritätsvorteile gemäss Abschnitt 35 der Zivilprozessordnung der Vereinigten Staaten, Paragraph 119 aller unten angegebenen Auslandsanmeldungen für ein Patent oder eine Erfindersurkunde, und habe auch alle Auslandsanmeldungen für ein Patent oder eine Erfindersurkunde nachstehend gekennzeichnet, die ein Anmeldedatum haben, das vor dem Anmeldedatum der Anmeldung liegt, für die Priorität beansprucht wird.

As a below named inventor, I hereby declare that:

My residence, post office address and citizenship are as stated below next to my name,

I believe I am the original, first and sole inventor (if only one name is listed below) or an original, first and joint inventor (if plural names are listed below) of the subject matter which is claimed and for which a patent is sought on the invention entitled

NETWORK, INTERPRETER FOR SUCH A NETWORK, AND METHOD FOR OPERATING A NETWORK

the specification of which

(check one)

☐ is attached hereto

☒ was filed on 12 May 2000
as United States application Number or PCT international application Number PCT/EP00/04312

and was amended on _____
(if applicable)

I hereby state that I have reviewed and understand the contents of the above identified specification, including the claims as amended by any amendment referred to above.

I acknowledge the duty to disclose information which is material to the examination of this application in accordance with Title 37, Code of Federal Regulations, §1.56.
I hereby claim foreign priority benefits under Title 35, United States Code, §119 of any foreign application(s) for patent or inventor's certificate listed below and have also identified below any foreign application for patent or inventor's certificate having a filing date before that of the application on which priority is claimed:

German Language Declaration

Prior foreign applications
Priorität beansprucht

			Priority Claimed	
199 22 118.9	Germany	12 May 1999	☒	☐
(Number)	(Country)	(Day Month Year Filed)	Yes	No
(Nummer)	(Land)	(Tag Monat Jahr eingereicht)	Ja	Nein
			☐	☐
(Number)	(Country)	(Day Month Year Filed)	Yes	No
(Nummer)	(Land)	(Tag Monat Jahr eingereicht)	Ja	Nein
			☐	☐
(Number)	(Country)	(Day Month Year Filed)	Yes	No
(Nummer)	(Land)	(Tag Monat Jahr eingereicht)	Ja	Nein

Ich beanspruche hiermit germäss Absatz 35 der Zivilprozessordnung der Vereinigten Staaten, Paragraph 120, den Vorzug aller unten aufgeführten Anmeldungen und falls der Gegenstand aus jedem Anspruch dieser Anmeldung nicht in einer früheren amerikanischen Patentanmeldung laut dem ersten Paragraphen des Absatzes 35 der Zivilprozeßordnung der Vereinigten Staaten, Paragraph 122 offenbart ist, erkenne ich gemäss Absatz 37, Bundesgesetzbuch, Paragraph 1.56 meine Pflicht zur Offenbarung von Informationen an, die zwischen dem Anmeldedatum der früheren Anmeldung und dem nationalen oder PCT internationalen Anmeldedatum dieser Anmeldung bekannt geworden sind.

I hereby claim the benefit under Title 35, United States Code. §120 of any United States application(s) listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of Title 35, United States Code, §122 I acknowledge the duty to disclose material information as defined in Title 37, Code of Federal Regulations, §1.56 which occurred between the filing date of the prior application and the national or PCT international filing date of this application.

(Application Serial No.) (Anmeldeseriennummer)	(Filing Date) (Anmeldedatum)	(Status) (patentiert, anhängig, aufgegeben)	(Status) (patented, pending, abandoned)
(Application Serial No.) (Anmeldeseriennummer)	(Filing Date) (Anmeldedatum)	(Status) (patentiert, anhängig, aufgegeben)	(Status) (patented, pending, abandoned)

Ich erkläre hiermit, dass alle von mir in der vorliegenden Eklärung gemachten Angaben nach meinem besten Wissen und Gewissen der vollen Wahrheit entsprechen, und dass ich diese eidesstattliche Erklärung in Kenntnis dessen abgebe, dass wissentlich und vorsätzlich falsche Angaben gemäss Paragraph 1001, Absatz 18 der Zivilprozessordnung der Vereinigten Staaten von Amerika mit Geldstrafe belegt und/oder Gefängnis bestraft werden koennen, und dass derartig wissentlich und vorsätzlich falsche Angaben die Gültigkeit der vorliegenden Patentanmeldung oder eines darauf erteilten Patentes gefährden können.

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true, and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

German Language Declaration

VERTRETUNGSVOLLMACHT: Als benannter Erfinder beauftrage ich hiermit den nachstehend benannten Patentanwalt (oder die nachstehend benannten Patentanwälte) und/oder Patent-Agenten mit der Verfolgung der vorliegenden Patentanmeldung sowie mit der Abwicklung aller damit verbundenen Geschäfte vor dem Patent- und Warenzeichenamt: (Name und Registrationsnummer anführen)

POWER OF ATTORNEY: As a named inventor, I hereby appoint the following attorney(s) and/or agent(s) to prosecute this application and transact all business in the Patent and Trademark Office connected therewith. (list name and registration number)

And I hereby appoint all Attorneys Identified by United States Patent & Trademark Office Customer Number 26574, who are all members of the firm of Schiff Hardin and Waite.

Telephone 312/-258-5500 Patent Department

my attorneys with full power of substitution and revocation, to prosecute this application and to transact all business in the Patent and Trademark Office connected therewith and direct that all correspondence be forwarded to:

Schiff, Hardin & Waite
Atten: Patent Department
6600 Sears Tower, Chicago, Illinois 60606 -6473
Customer Number 26574

Voller Name des einzigen oder ursprünglichen Erfinders: 1 - 00	Full name of sole or first inventor: ANDREAS HOFSTETTER
Unterschrift des Erfinders Datum	Inventor's signature Andreas Hofstetter Date 10/11/01
Wohnsitz	Residence D82024 Taufkirchen DEX Germany
Staatsangehörigkeit	Citizenship Germany
Postanschrift	Post Office Address Haselweg 5
	82024 Taufkirchen Germany